ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΣΧΕΔΙΑΣΗ ΤΡΙΣΔΙΑΣΤΑΤΩΝ ΠΑΙΧΝΙΔΙΩΝ ΜΕ ΤΗΝ ΑΝΟΙΧΤΗ ΣΟΥΙΤΑ BLENDER

Φοιτήτρια:
Τσαχειρίδου Μαρία
Αρ. Μητρώου: 07/3202

Επιβλέπων καθηγητής:
Κωνσταντίνος Διαμαντάρας

ΘΕΣΣΑΛΟΝΙΚΗ 2013

ΠΡΟΛΟΓΟΣ

Η πτυχιακή εργασία “Σχεδίαση Τρισδιάστατων Παιχνιδιών με την ανοιχτή σουίτα Blender” πραγματοποιήθηκε μέσα στο πλαίσιο των σπουδών μου στο τμήμα Πληροφορικής του ΤΕΙ Θεσσαλονίκης με υπεύθυνο και επιβλέποντα καθηγητή τον κύριο Κωνσταντίνο Διαμαντάρα.

Στην πολύμηνη αυτή εργασία, προσπάθησα να αποδώσω όσο καλύτερα γινόταν ένα κείμενο εκπαιδευτικού χαρακτήρα, σε όσους θέλουν να μάθουν τις δυνατότητες της σουίτας Blender γύρω από τον τομέα του Game Engine. Υλοποιήθηκαν δύο παιχνίδια εκ των οποίων το ένα αναλύεται σε αυτή την πτυχιακή.

Στο **1ο Κεφάλαιο** αναλύονται οι καινούργιες επιλογές που προστίθενται στο Blender με τη μετάβαση του σε Game Engine. Επεξηγούνται μαζί με λίγα λόγια για τον Logic Editor οποίος αναλύεται παρακάτω.

Στο **2ο Κεφάλαιο** περιγράφονται και οι τρεις πίνακες λογικών τούβλων. Σε κάθε μία κατηγορία(αισθητήρες, ελεγκτές, ενεργοποιητές), επεξηγούνται ένας προς ένας οι τύποι από τους οποίους αποτελούνται.

Στο **3ο Κεφάλαιο** δίνεται με βήματα η διαδικασία δημιουργίας του παιχνιδιού Turret Defense, το οποίο βοηθάει στην κατανόηση των κουμπιών και μενού του Game Engine.

Στο **4ο Κεφάλαιο** περιγράφονται οι πρόσθετες επιλογές του παιχνιδιού που κατασκευάζουμε. Δίνονται τα βήματα για την κατασκευή απλού animation, δυναμικού κειμένου, κώδικα Python αλλά και άλλων δυνατοτήτων του Blender

Στη συνέχεια δίνονται τα **συμπεράσματα** που βγήκαν στην προσπάθεια κατασκευής παιχνιδιών με τη συγκεκριμένη σουίτα.

Έπειτα ως **επίλογος** δίνονται οι νέες δυνατότητες του Blender που συνοδεύουν τις νέες εκδόσεις, καθώς και προτάσεις για τη βελτιστοποίηση του.

Τέλος παρατίθεται η **βιβλιογραφία** που βοήθησε σημαντικά στην κατανόηση τόσο του Blender όσο και της παιχνιδομηχανής του.

ΠΕΡΙΛΗΨΗ

Ανάμεσα στις μηχανές ανάπτυξης βιντεοπαιχνιδιών, είναι και η ανοιχτή σουίτα γραφικών Blender, η οποία διαθέτει δικιά της μηχανή ανάπτυξης τρισδιάστατων παιχνιδιών. Η μηχανή αυτή βασίζεται στη λογική του παιχνιδιού. Με τα ειδικά πάνελ, μενού και κουμπιά, ο κάθε χρήστης μπορεί να διαχειριστεί τα λογικά τούβλα, όπως ονομάζονται, και να προσθέσει κίνηση στα αντικείμενα καθώς και άλλες δράσεις που μπορούν να συντελέσουν μέρος της δημιουργίας ενός παιχνιδιού. Έχει τη δυνατότητα να «παίξει» με τις ιδιότητες και τη φυσική του κάθε αντικειμένου, καθώς και να προγραμματίσει σε κώδικα Python. Ένα τέτοιο παιχνίδι είναι και το Turret Defense, το κανόνι στη μέση της σκηνής που προσπαθεί να αντικρούσει τους εχθρούς που τον πλησιάζουν. Αναπτύσσεται βήμα προς βήμα, σαν εργαλείο εκμάθησης του Blender από τον αρχάριο χρήστη. Παρόλα τα προβλήματα που μπορεί να αντιμετωπίσει ο σχεδιαστής, το μέλλον για τη σουίτα Blender διαγράφεται λαμπρό, αφού ήδη έχει να επιδείξει μια σημαντική γκάμα animation και παιχνιδιών δημιουργημένα από τους χρήστες της.

ABSTRACT

Among the engines for game developing, we find the open graphics suite Blender, which features its own 3d game development engine. This engine is based on the logic of the game. With the special panels, menus and buttons, each user can manage the logic bricks, as they are called, and add movement to objects, as well as other actions that can contribute as a part of the game creation. He has the ability of 'playing' with the properties and physics of each object and programming in Python code. Turret Defense is a game like this, with the cannon in the middle of the scene trying to refute the enemies that want to reach it. It is developed step by step, as a tool of learning Blender, for every new user. Despite the problems, that any designer may face, the future of Blender suite is bright, having already an important range of animation and games to demonstrate, created from its users.

ΕΥΧΑΡΙΣΤΙΕΣ

Ένα μεγάλο ευχαριστώ επιβάλλεται να δοθεί στον επιβλέποντα καθηγητή, κύριο Κωνσταντίνο Διαμαντάρα, πρώτα από όλα για την αστείρευτη υπομονή που έδειξε όλους αυτούς τους μήνες έρευνας και ανάπτυξης της πτυχιακής εργασίας. Με την θετική διάθεση που έδειξε και την άμεση ανταπόκρισή του σε κάθε πρόβλημα που παρουσιάστηκε, βοήθησε στην σωστή και έγκαιρη ολοκλήρωση της εργασίας, δίνοντας συνεχώς τροφή για περαιτέρω έρευνα με αποτέλεσμα την καλύτερη εκμάθηση της σουίτας Blender.

Έπειτα θα ήθελα να ευχαριστήσω τους γονείς μου, που φρόντισαν όλα αυτά τα ιδιαίτερα χρόνια των σπουδών, να με στηρίξουν με κάθε τρόπο δείχνοντας την απαραίτητη υπομονή και δίνοντας σωστή και συνεχή καθοδήγηση.

ΠΕΡΙΕΧΟΜΕΝΑ

Ευρετήριο σχημάτων

ΕΙΣΑΓΩΓΗ ΣΤΙΣ 3D ΜΗΧΑΝΕΣ ΑΝΑΠΤΥΞΗΣ ΠΑΙΧΝΙΔΙΩΝ (GAME ENGINES)

Στον κόσμο των 3d γραφικών Η/Υ η ανάγκη για δημιουργία δεν σταματάει στην παραγωγή εικόνας και animation. Με την τεχνολογία να προχωράει με εκθετικούς ρυθμούς, ο μέσος χρήστης επιζητά πολύ συχνά την ψυχαγωγία του στον φανταστικό κόσμο των παιχνιδιών μέσω υπολογιστή ή κάποιας από τις κονσόλες ηλεκτρονικών παιχνιδιών. Ο προγραμματιστής των παιχνιδιών αυτών, είναι επομένως υποχρεωμένος να ακολουθεί και να ικανοποιεί τις απαιτήσεις του παίκτη, δημιουργώντας όλο και πιο αληθοφανή γραφικά με κάποια από τις μηχανές ανάπτυξης παιχνιδιών που κυκλοφορούν.

Μια μηχανή ανάπτυξης παιχνιδιών αποτελείται από ένα περιβάλλον που παρέχει στον χρήστη της πλήρη γκάμα κουμπιών και μενού για την εύκολη, γρήγορα και αποδοτική δημιουργία παιχνιδιών. Παρέχει όλη την αρχιτεκτονική στήριξη από την ανάπτυξη μέχρι και την εξαγωγή ενός παιχνιδιού που μπορεί να σταθεί μόνο του.

Ένας σχεδιαστής στην σημερινή εποχή θα αναζητήσει χαρακτηριστικά για το έργο του που πρέπει να δίνονται με εύκολο και γρήγορο τρόπο χωρίς να είναι αναγκασμένος να τα δημιουργήσει από την αρχή. Τέτοια χαρακτηριστικά είναι η μηχανή της φυσική υπόστασης των αντικειμένων, ο σχεδιασμός τους, η κίνησής τους, ο ήχος αλλά και η νοημοσύνη τους.

Αρκετά σημαντικός τομέας είναι και το πλαίσιο του προγραμματισμού. Θα επιλεγεί φυσικά η σουίτα με την αποδοτικότερη γλώσσα προγραμματισμού, η οποία θα του προσφέρει την δυνατότητα συγκεκριμενοποίησης των στοιχείων του παιχνιδιού, ακριβώς με τον τρόπο που τα ζητάει ο παίκτης ή ο ίδιος ο σχεδιαστής.

Υπάρχει φυσικά μια ευρεία γκάμα λογισμικού για το συγκεκριμένο αντικείμενο. Πολλές σουίτες γραφικών διαθέτουν τις δικιές τους μηχανές, ενώ άλλες αποτελούν αυτοτελή κομμάτια για την ανάπτυξη παιχνιδιών. Παρακάτω θα παρατεθούν κάποιες από τις πιο δημοφιλείς καθώς και μια πρώτη ματιά γύρω από τη χρήση τους.

UNREAL ENGINE

Figure 1: **Unreal Engine**

Αποτελεί μια πλήρη εργαλειοθήκη για δημιουργία βιντεοπαιχνιδιών. Υπάρχει από το 1998 και ανήκει στην εταιρεία ανάπτυξης βιντεοπαιχνιδιών EPIC GAMES. Μερικά από τα πιο διάσημα παιχνίδια είναι αποτέλεσμα σχεδίασης πάνω στην συγκεκριμένη μηχανή(Batman: Arkham City, Gears of War Series και άλλα). Αν και η πλήρης σουίτα προσφέρεται έναντι αμοιβής και μόνο για επαγγελματίες σχεδιαστές παιχνιδιών, υπάρχει η έκδοση UDK που είναι ανοιχτή προς το κοινό για εκπαιδευτικούς κυρίως λόγους. Δίνει παράλληλα όμως και την ευκαιρία για παραπάνω ανάπτυξη και πώληση του τελικού προιόντος μέσω της διαφημιστικής άδειας που παρέχει.

CRY ENGINE

Figure 2 : **Cry Engine**

Η διαδεδομένη μηχανή CRY ENGINE αποτελεί και αυτή με τη σειρά της ένα πολύ δυνατό εργαλείο για την ανάπτυξη παιχνιδιών. Ανήκει στην εταιρεία CRYTEK και αναπτύχθηκε την περίοδο 2001-2004. Φημίζεται για την δυνατότητα δημιουργίας τοπίων με υψηλό βαθμό λεπτομέρειας. Αν και οι περισσότερες εκδόσεις της είναι επί πληρωμής και δύσκολο να αποκτηθούν, υπάρχει και σε αυτή την περίπτωση η έκδοση SDK για ελεύθερη χρήση.

UNITY 3D

Figure 3 : **Unity**

Η Unity είναι μια μηχανή παιχνιδιών που μπορεί να λειτουργήσει σε διάφορες πλατφόρμες. Ανήκει στης UNITY TECHNOLOGIES και τα παιχνίδια της είναι κατάλληλα τόσο για υπολογιστή, όσο και για κονσόλες και λειτουργικά κινητών τηλεφώνων. Παρέχει πλήρη γκάμα εργαλείων για ανάπτυξη 3d και 2d γραφικών για παιχνίδια. Θεωρείται μια εύκολη ως προς τη χρήση μηχανή, κατάλληλη για νέους χρήστες. Η στάνταρ έκδοσή της είναι δωρεάν αλλά υπάρχει και η έκδοση Unity Pro με υψηλά χαρακτηριστικά για επαγγελματίες του είδους.

PANDA 3D

Είναι μια δημιουργία του Disney VR Studio και η πρώτη κυκλοφορία έγινε το 2002. Κατά βάση είναι μια μηχανή για γραφικά σκηνής και προορίζεται για ελεύθερη χρήση. Δίνει τη δυνατότητα δημιουργίας 3d παιχνιδιών με γλώσσα προγραμματισμού την Python αλλά και τη C++. Η κοινωνία του Panda 3d δεν είναι μεγάλη αλλά έχει ήδη αναπτύξει ένα σύνολο από διαφημιστικά παιχνίδια και projects με πολύ συγκεκριμένο ύφος και χαρακτηριστικά γραφικά.

Figure 4 : Panda 3d Image

Πολλές από τις μηχανές παιχνιδιών που κυκλοφορούν σήμερα στο εμπόριο, υπερτερούν του Blender, κυρίως στον τομέα των γραφικών αλλά και γύρω από τις δυνατότητες που προσφέρουν στον σχεδιαστή/προγραμματιστή. Οι λόγοι που προτιμήθηκε η σουίτα Blender για τη συγκεκριμένη πτυχιακή ξεκινούν αρχικά από το γεγονός ότι είναι ελεύθερη ως προς τον κάθε χρήστη. Είδαμε φυσικά πως και οι υπόλοιπες μηχανές διαθέτουν κάποια δωρεάν έκδοση, το πλήρες πακέτο όμως προσφέρεται έναντι αμοιβής σε επαγγελματίες σχεδιαστές. Αυτό πρακτικά σημαίνει πως ο απλός χρήστης δεν μπορεί να αξιοποιήσει το 100% των δυνατοτήτων του. Στον αντίποδα βρίσκεται το Blender προσφέροντας σε κάθε χρήστη ολόκληρο το πακέτο των κουμπιών και μενού από το οποίο αποτελείται, δίνοντάς του την ευκαιρία για ολοκληρωμένη σχεδίαση.

Αν και η σουίτα Blender είναι εδραιωμένη κυρίως στον χώρο των Animation, έχοντας αποδείξει πως μπορεί να σταθεί επάξια, απέναντι σε άλλες δυνατές μηχανές, θέλω να δώσω την ευκαιρία ενασχόλησης και με την μηχανή παιχνιδιών, ένα αναπτυσσόμενο ακόμη στοιχείο της, που προβλέπεται πως μελλοντικά θα κατακτήσει τον χώρο των βιντεοπαιχνιδιών. Ας μην ξεχνάμε πως για τον αρχάριο χρήστη προσφέρει ένα εύκολο για εκμάθηση περιβάλλον, παρόλη την ποικιλία κουμπιών που διαθέτει. Με εξάσκηση λίγων εβδομάδων ο καθένας θα μπορεί να βρίσκεται σε θέση να αναπτύξει ένα απλό παιχνίδι, με βάση την λογική και τους κανόνες της φυσικής.

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ ΣΤΟ BLENDER GAME ENGINE

Το Blender είναι μια σουίτα ανοιχτού λογισμικού, δωρεάν, για τη δημιουργία 3D γραφικών υπολογιστή. Χρησιμοποιείται για τη δημιουργία οπτικών εφέ, για animation, για εικόνες, για προσομοιώσεις και για έναν ακόμη τομέα με τον οποίο θα ασχοληθούμε στην τρέχουσα πτυχιακή εργασία, την δημιουργία Βιντεοπαιχνιδιών. Η τελευταία έκδοση του Blender ανακοινώθηκε στις 7 Μαΐου 2013 και είναι η 2.67 .

Η ανάπτυξη βιντεοπαιχνιδιών αποτελεί ένα υπό-έργο του Blender και προσφέρει κυρίως την αλληλεπίδραση μεταξύ αντικειμένων στη σκηνή. Ασχολείται με συγκρούσεις αντικειμένων, ανίχνευση αυτών που βρίσκονται κοντά στα επιλεγμένα, αλλαγές στη φυσική των αντικειμένων, επίσης με τον προγραμματισμό συγκεκριμένης λογικής με δικό του πάνελ αλλά και τον κινητήρα που αφορά τη δυναμική του παιχνιδιού. Εκτός των άλλων δίνει τη σημαντική δυνατότητα δημιουργίας stand-alone παιχνιδιού με τη συνοδεία των κατάλληλων dll αρχείων.

Η παιχνιδομηχανή είναι ενσωματωμένη στην βάση κώδικα του Blender. Είναι γραμμένη εξολοκλήρου σε C++ και λειτουργεί ως ανεξάρτητο στοιχείο. Στην εναλλαγή από το απλό σύστημα Blender σε Game Engine η διάταξη οθόνης αλλάζει σε κάποια σημεία και προστίθενται κάποια χαρακτηριστικά modeling, κουμπιά και πάνελ κατάλληλα για τη δημιουργία και λειτουργία αυτόνομου παιχνιδιού Blender. Χρησιμοποιεί OpenGL μια πολύ-πλατφόρμα API για την επικοινωνία με το hardware γραφικών.

Όσον αφορά την ιστορία της, αναπτύχθηκε για πρώτη φορά το 2000 από τους Erwin Coumans και Gino van den Bergen, προκειμένου να δημιουργηθεί μια πλατφόρμα φιλική προς τον χρήστη για εύκολη δημιουργία παιχνιδιών. Μέχρι την έκδοση 2.37a του Blender, η παιχνιδομηχανή δεν λειτουργούσε λόγω μη ανοιχτού λογισμικού της βιβλιοθήκης που αφορούσε τη φυσική. Η έκδοση 2.42 ήταν αυτή στην οποίο φάνηκε η πρώτη σημαντική πρόοδος.

Η διαδικασία rendering διαφέρει σημαντικά με αυτής του απλού συστήματος Blender. Όταν μιλάμε για animation ή απλές εικόνες σε χρόνο rendering η εικόνα δεν μπορεί να τροποποιηθεί. Για να συμβεί αυτό πρέπει να βρισκόμαστε σε mode επεξεργασίας από τον χρήστη. Αντίθετα στο Game Engine οι σκηνές μπορούν να αλλάξουν σε πραγματικό χρόνο και να τροποποιηθούν είτε μόνες τους είτε από το χρήστη, εφόσον έχουμε θέσει τις κατάλληλες λειτουργίες στα αντικείμενα. Για παράδειγμα σε πραγματικό χρόνο ο χρήστης μπορεί να κουνήσει ένα αντικείμενα και έτσι να αλλάξει τη θέση του στη σκηνή.

Το γραφικό περιβάλλον της σουίτας Blender θεωρείται πολύπλοκο λόγω της ποικιλίας buttons και των panels. Οι δυνατότητες όμως είναι απεριόριστες και πολλοί χρήστες, ακόμη και αρχάριοι το προτιμούν αφού δίνει τη δυνατότητα

δημιουργίας περιβάλλοντος, κίνησης, χαρακτήρων και πολλών άλλων, τόσο με τα ενσωματωμένα εργαλεία, όσο και με τον προγραμματισμό με τη γλώσσα Python. Τέλος το Blender περιλαμβάνει δικό του οδηγό χρήσης, χρήσιμο για τον αρχάριο κυρίως χρήστη, ως προς την εκμάθηση των δυνατοτήτων της σουίτας, στην σελίδα http://wiki.blender.org/

1.1 ΑΛΛΑΓΕΣ ΟΘΟΝΗΣ ΣΤΗ ΜΕΤΑΒΑΣΗ ΣΕ BLENDER GAME

Η Μηχανή του Blender για παιχνίδι διευκολύνει τον χρήστη δίνοντας του κάποιες δυνατότητες που προσφέρονται αποκλειστικά για αυτό τον σκοπό. Η οθόνη και η διάταξή της συνεχίζουν να έχουν τα μενού που προϋπήρχαν στο Blender Render, αλλά με τη μετάβαση στην μηχανή παιχνιδιού προστίθενται κάποια καινούργια χαρακτηριστικά τα οποία θα αναλυθούν στη συνέχεια.

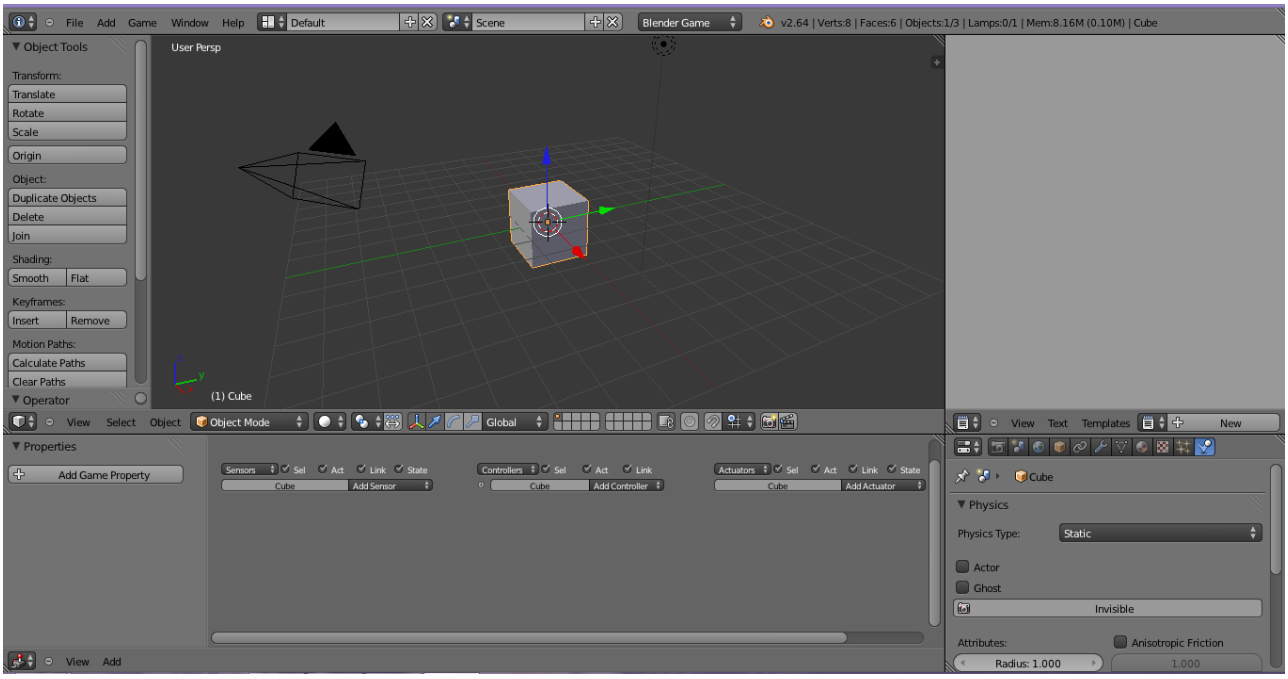

Figure 5 : 1.1.1 Game Logic Screen Layout

1. Στο μενού που βρίσκεται στην κορυφή της οθόνης του Blender υπάρχει το πεδίο που αφορά τη διευκρίνιση σχετικά με το ποια μηχανή του Blender θα χρησιμοποιηθεί για την απόδοση της εικόνας σε πραγματικό χρόνο(render). Η μετάβαση σε Blender Game είναι επομένως ένα αρχικό βήμα. Με την επιλογή αυτή ενεργοποιούνται μενού που δεν ήταν ορατά με τις άλλες μηχανές.
2. Στο μενού στην κορυφή της οθόνης του Blender και δίπλα από τις επιλογές File και Add βρίσκεται η επιλογή Game με το αναδυόμενο μενού της. Η επιλογή αυτή είναι ορατή εφόσον έχουμε μεταβεί σε Blender Game Engine.

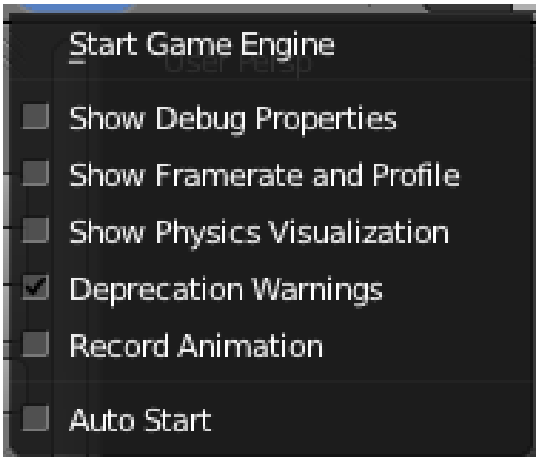

Figure 6 : 1.1.2 Μενού Game

Μενού της επιλογής Game:

Start Game Engine – Με τη συγκεκριμένη επιλογή το παιχνίδι τίθεται σε λειτουργία (Για συντόμευση χρησιμοποιείται το πλήκτρο P).

Show Debug Properties – Κατά τη διάρκεια δημιουργίας του παιχνιδιού ίσως χρειαστεί να προσθέσουμε κάποιες ιδιότητες. Σε αυτές θα τεθούν τιμές που σε ορισμένες περιπτώσεις θα αλλάζουν καθώς το παιχνίδι τρέχει. Σε περίπτωση που θέλουμε να ελέγχουμε τις αλλαγές αυτές, τις μαρκάρουμε για debugging και έπειτα μαρκάρουμε και τη συγκεκριμένη επιλογή.

Show framerate and Profile – Εφόσον επιλεγεί, θα εμφανίζονται πληροφορίες τόσο των frames όσο και των:
Swap, Physics ,Logic, Animations, Network, Scenegraph, Rasterizer, Services, Overhead, Outside
καθώς το παιχνίδι τρέχει. Το layout δεν είναι τίποτα παρά στατιστικά στοιχεία για το κάθε ένα από αυτά που αφορούν τον χρόνο που ξοδεύεται για την απόδοσή τους. Επίσης δίνει τα frames per second που χρησιμοποιούνται σε μέτρηση %.

Show Physics visualization: Πολύ χρήσιμη επιλογή όταν θέλουμε να ελέγξουμε τα όρια των αντικειμένων, ή τους περιορισμούς που θέσαμε ή ακόμη και τις αλληλεπιδράσεις μεταξύ τους. Για παράδειγμα αν έχουμε ένα αόρατο αντικείμενο, αλλά πρέπει να ξέρουμε ότι δουλεύει σωστά, η συγκεκριμένη επιλογή θα το κάνει κατά κάποιο τρόπο ορατό ώστε να διορθώσουμε τυχόν λάθη.

Deprecation Warnings: Τυπώνει προειδοποιήσεις που αφορούν χαρακτηριστικά γνωρίσματα υποτιμημένα στο Python API.

Record animation: Καταγράφει animation την ώρα που τρέχει το παιχνίδι.

Auto Start: Ξεκινάει το παιχνίδι αυτόματα σε χρόνο φόρτου.

3. Ο Logic Editor είναι το πιο σημαντικό πάνελ όσον αφορά την δημιουργία παιχνιδιού στο Blender. Δημιουργεί κίνηση, αντιδράσεις, προσθέτει ήχο αλλά και πολλά άλλα στα αντικείμενα που χρησιμοποιούνται. Είναι υπεύθυνο για το μεγαλύτερο μέρος της αλληλεπίδρασης στο παιχνίδι. Θα αναλυθεί περαιτέρω στο Κεφάλαιο: Logic Editor.
4. Ιδιότητες-Properties. Στο δεξί κομμάτι του πάνελ Logic Editor βρίσκουμε το παράθυρο των ιδιοτήτων, το οποίο αφορά μεταβλητές που θέτει ο χρήστης στα αντικείμενα για διάφορες εργασίες(απόκτηση, αλλαγή τιμών κτλ). Θα αναλυθεί περαιτέρω στο Κεφάλαιο: Ιδιότητες-Properties.

1.2 LOGIC EDITOR – ΛΙΓΑ ΛΟΓΙΑ

Ο Logic Editor είναι ένα ξεχωριστό πάνελ με ιδιαίτερα σημασία στο BGE. Ασχολείται με την λογική του παιχνιδιού και την αναπαριστά ως λογικά τούβλα. Τα περισσότερα αντικείμενα στην δημιουργία ενός παιχνιδιού έχουν κάποια λειτουργία. Ορισμένα μετακινούνται συνεχώς ή υπό ορισμένες συνθήκες, άλλα αλλάζουν μορφή ή αλληλεπιδρούν με τα γειτονικά τους. Υπάρχουν συγκρούσεις αλλά και αλλαγές σκηνής. Αυτά είναι μόνο λίγα από τα παραδείγματα λογικής που διαχειρίζεται ο Logic Editor.

Για να λειτουργήσει η λογική γίνεται σύνδεση των λογικών τούβλων. Λογικά τούβλα(bricks), είναι ένας πίνακας με τρεις στήλες. Οι στήλες αυτές ονομάζονται Αισθητήρες(Sensors), Ελεγκτές(Controllers) και Ενεργοποιητές(Actuators). Οι τρεις αυτές στήλες αποτελούνται από επιλογές που τελούν διαφορετικές λειτουργίες.

Συνδέονται μεταξύ τους και με αυτόν τον τρόπο θετικά σήματα μεταδίδονται από τη μια στήλη στην άλλη, ενεργοποιώντας τις λειτουργίες της κάθε στήλης και δίνοντας ένα τελικό αποτέλεσμα(κάποια αλληλεπίδραση ή αλλαγή αντικειμένου/σκηνής). Οι συνδέσεις μπορούν να συμβούν στους παλμούς(σήματα) μεταξύ Αισθητήρα-Ελεγκτή ή Ελεγκτή-Ενεργοποιητή και όχι διαφορετικά.

Οι γραμμές συνδέσεων δημιουργούνται μετά από σύρσιμο του ποντικιού από τις μαύρες τελείες του ενός κόμβου στον άλλον. Αν και όπως είπαμε η σύνδεση έχει ορισμένη ροή, εντούτοις είναι εφικτό ένας Αισθητήρας να ενωθεί με πολλαπλούς Ελεγκτές, αλλά και ένα Ελεγκτής να ενωθεί με πολλαπλούς Ενεργοποιητές. Αντίστοιχα οι κόμβοι υποδοχής δέχονται πολλαπλές συνδέσεις. Η διαγραφή μιας σύνδεσης γίνεται και πάλι με σύρσιμο της γραμμής μεταξύ των κόμβων.

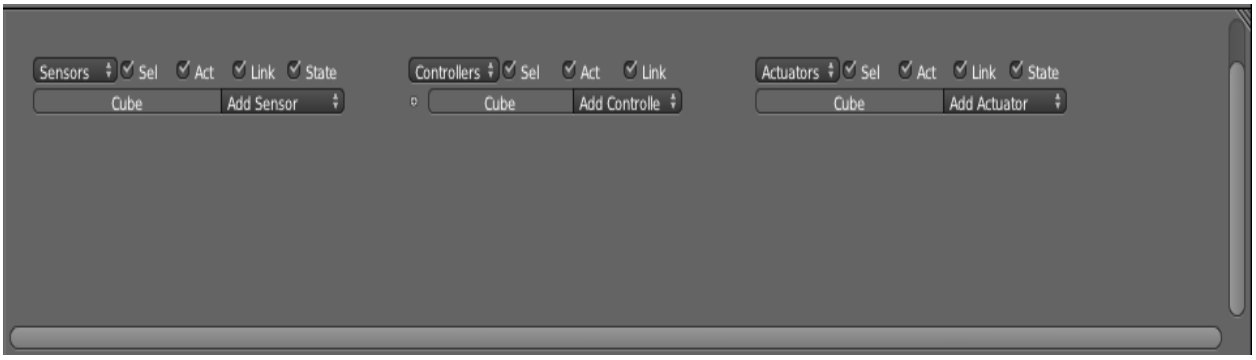

Figure 7 : 1.2.1 Logic Editor-Logic Bricks

1.3 ΙΔΙΟΤΗΤΕΣ – PROPERTIES

Το πάνελ των Ιδιοτήτων βρίσκεται στα αριστερά του Logic Editor και αφορά και πάλι τα αντικείμενα του παιχνιδιού. Είναι άμεσα συνδεδεμένο με τα λογικά τούβλα, αφού χρησιμοποιείται για πολλές διεργασίες.

Στο πάνελ Ιδιοτήτων συναντάμε αρχικά το κουμπί Add Game Property. Με το πάτημα του εισάγεται μια νέα ιδιότητα. Η ιδιότητα αυτή είναι μια μεταβλητή με τύπο και προορίζεται για συγκεκριμένη χρήση. Αρχικοποιείται και μπορούν αν ρυθμιστούν τη στιγμή που το παιχνίδι τρέχει με τον κατάλληλο Sensor και Actuator.

Υπάρχουν πέντε τύποι Ιδιοτήτων:

Timer	Ξεκινά με την τιμή της ιδιότητας και μετράει προς τα πάνω όσο το αντικείμενο υφίσταται. Μπορεί παραδείγματος χάριν να χρησιμοποιηθεί εάν θέλετε να ξέρετε πόσο χρόνο θα πάρει για έναν παίκτη να ολοκληρώσει ένα επίπεδο.
Float	Χρησιμοποιεί δεκαδικούς αριθμούς ως τιμές, που κυμαίνονται από -10000.000 μέχρι 10000.000. Είναι χρήσιμο για τις τιμές ακριβείας.
Integer	Χρησιμοποιεί integers (ακέραιους αριθμούς) ως τιμές, μεταξύ -10000 και 10000. Χρήσιμο για καταμέτρηση πραγμάτων όπως πυρομαχικά, όπου οι δεκαδικοί δεν είναι χρήσιμοι.
String	Λαμβάνει το κείμενο ως τιμή. Μπορεί να αποθηκεύσει 128 χαρακτήρες.
Boolean	Η μεταβλητή Boolean, έχει δυο τιμές: true ή false. Αυτό είναι χρήσιμο για πράγματα που έχουν μόνο δυο λειτουργίες.

1.3.1 ΕΙΔΙΚΕΣ ΕΠΙΛΟΓΕΣ ΙΔΙΟΤΗΤΑΣ

Πεδίο Name:

Το όνομα της ιδιότητας. Με αυτό το όνομα θα έχουν πρόσβαση σε αυτήν, οι εκφράσεις ή ο κώδικας Python. Γίνεται διάκριση μεταξύ πεζών και κεφαλαίων.

Type:

Ανοίγει τον κατάλογο των τύπων της ιδιότητας. Ο χρήστης θα διαλέξει αυτόν που του αρμόζει για την ιδιότητα που έθεσε.

Πεδίο Value:

Ο χρήστης θέτει μια αρχική τιμή για την ιδιότητα. Μπορεί να παραμείνει μηδέν. Στην περίπτωση Boolean θα είναι True ή False.

Κουμπί i:

Αν για οποιονδήποτε λόγο θέλουμε την ώρα που τρέχει το παιχνίδι να παρατηρούμε τις αλλαγές στην μεταβλητή, τότε επιλέγουμε το συγκεκριμένο κουμπί. Εφόσον θέσουμε και το debugging on στο Show Debug Properties από το μενού Game, θα έχουμε προβολή του ονόματος της ιδιότητας, του αντικειμένου που την κατέχει αλλά και της εκάστοτε τιμή της.

Κουμπί X:

Διαγράφει την ιδιότητα.

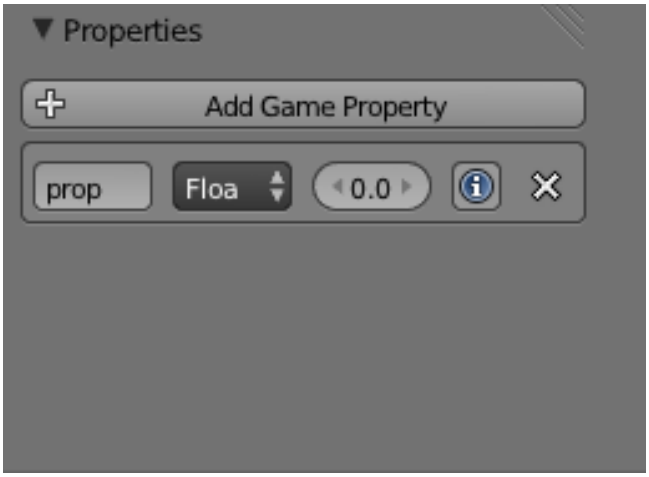

Figure 8 : 1.3.1.1 Game Property Area

1.4 ΚΑΤΑΣΤΑΣΕΙΣ(STATES)

Το σύστημα των καταστάσεων ανήκει στην Λογική του Παιχνιδιού και ενσωματώνεται στους Ελεγκτές(Controllers) των λογικών τούβλων. Αποτελεί την μικρή κουκίδα αριστερά της στήλης των ελεγκτών και για να ενεργοποιηθεί αρκεί κάποιος να πατήσει επάνω της. Με τον τρόπο αυτό γίνονται ορατές δύο νέες περιοχές, η 'Visible' και η 'Initial'.

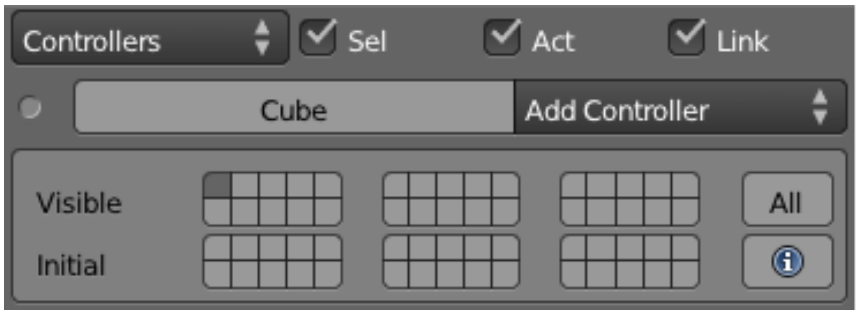

Figure 9 : **1.4.1 Καταστάσεις(States)**

Οι καταστάσεις αφορούν τα αντικείμενα. Κάθε αντικείμενο μπορεί να βρίσκεται σε μια ορισμένη μόνο κατάσταση κάθε φορά. Υπάρχουν 30 καταστάσεις για κάθε ένα object της σκηνής του σχεδιαστή. Σε κάθε μία είναι δυνατό να υπάρχει μια διαφορετική συμπεριφορά η οποία εκδηλώνεται με την αλλαγή της μιας κατάστασης σε μια άλλη.

Η περιοχή 'Visible' αφορά τις καταστάσεις που θέλουμε να χρησιμοποιήσουμε και όποια κουτιά είναι πιο σκούρα, τότε αυτά είναι τα τρέχοντα. Για να κατανοηθεί καλύτερα η χρήση τους θα δοθεί ένα παράδειγμα. Έχουμε επομένως την κίνηση ενός κύβου προς τα δεξιά με το πάτημα του πλήκτρου με το δεξί βελάκι. Θέλουμε έπειτα από τη σύγκρουση του κύβου με ένα αντικείμενο του χώρου το δεξί βελάκι να οδηγεί τον κύβο αριστερά. Στην πρώτη φυσικά κατάσταση θα έχουμε ενώσει τα κατάλληλα λογικά τούβλα για την δημιουργία της ορθής κίνησης. Στο σημείο αυτό επιλέγουμε την δεύτερη κατάσταση και ξαναεπιλέγουμε τα ίδια λογικά τούβλα με πριν, με μόνη διαφορά την αλλαγή κίνησης στον Ενεργοποιητή Motion, προς τα αριστερά. Τη στιγμή της σύγκρουσης ορίζουμε τον ενεργοποιητή State και φυσικά την δεύτερη κατάσταση ώστε να μεταβούμε και σε αυτήν.

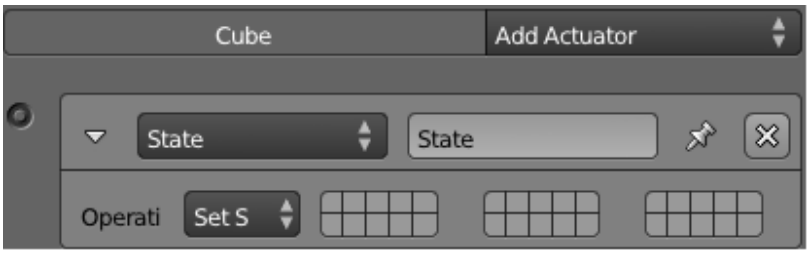

Figure 10 : **1.4.2 Ενεργοποιητής State**

Η περιοχή ‘Initial’ αφορά την κατάσταση που θα επιλέξει ο σχεδιαστής για να είναι η αρχική του, όταν το παιχνίδι ξεκινάει.

1.5 ΦΥΣΙΚΗ(PHYSICS)

Το Blender περιλαμβάνει προσομοίωση φυσικής για τα αντικείμενα της σκηνής. Με το ειδικό πάνελ που διαθέτει δίνει την δυνατότητα στον χρήστη, να προσθέσει τα κατάλληλα φυσικά γνωρίσματα στα αντικείμενά του, χωρίς να χρειαστεί περαιτέρω προγραμματισμός, αφού το ίδιο το Blender θα αναλάβει τη συνέχεια.

Για να βρούμε τις ιδιότητες για τα Φυσικά γνωρίσματα πρέπει να κάνουμε κάποια απλά βήματα. Πρώτα από όλα αλλάζουμε τη μηχανή από Blender Render σε Blender Game.

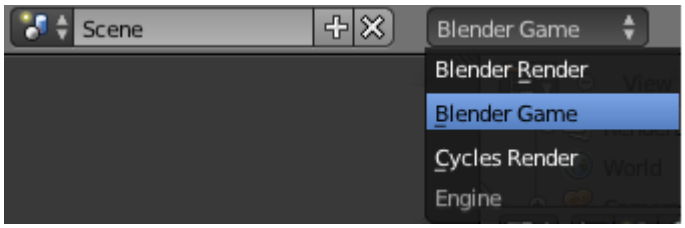

Figure 11 : **1.5.1 Αλλαγή σε Blender Game**

Στα κουμπιά του Editor των Ιδιοτήτων επιλέγουμε την καρτέλα Physics.

Figure 12 : **1.5.2 Properties Editor**

Η καρτέλα που εμφανίζεται δίνει τις επιλογές για διαχείριση των Physics, Collision Bounds και Create Obstacle.

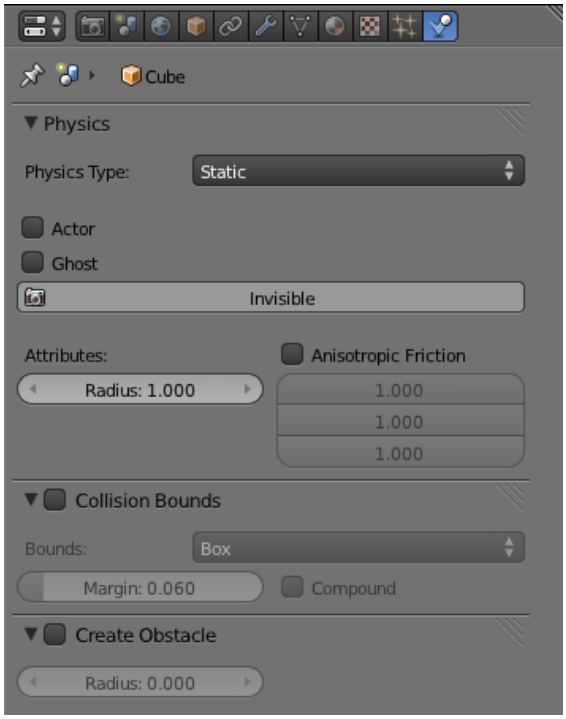

Figure 13 : **1.5.3 Physics panel**

Physics:

Physics Type-

Απευθύνεται στον τύπο που θα είναι το αντικείμενο όσον αφορά τη φυσική του συμπεριφορά στον χώρο. Αν επιλεγεί :

No Collision → Τότε δεν θα αλληλεπιδρά με κανένα αντικείμενο της σκηνής.

Static → Θα μπορεί απλά να καταλάβει τις συγκρούσεις και να ανταποκριθεί σαν να έχει έναν αισθητό όγκο.

Dynamic → Θα ανταποκρίνεται στις συγκρούσεις και θα έχει την ικανότητα της κίνησης. Θα πρέπει να στέκεται κάπου αλλιώς η βαρύτητα θα το τραβάει προς τα κάτω.

Rigid Body → Ίδιο με το Dynamic, με προσθήκη την ικανότητα περιστροφής.

Soft Body → Οι συνθήκες της σκηνής μπορούν εκτός των άλλων να το κάνουν να αλλάξει μορφή.

Άλλες επιλογές είναι η Actor που θέτει ένα αντικείμενο ορατό στους αισθητήρες Near και Sensor, αλλά και η Ghost που επιτρέπει το αντικείμενο να περνάει μέσα από άλλα.

Με το κουμπί Invisible θέτει το αντικείμενο αόρατο, όταν το παιχνίδι τρέχει.

Η επιλογή Anisotropic Friction αναφέρεται την αντίσταση που θα προβάλλεται σε ένα αντικείμενο με βάση κάποιον άξονα. Κάποια αντικείμενα στο φυσικό περιβάλλον κινούνται ευκολότερα σε κάποια μεριά και δυσκολότερα σε κάποια άλλη, όπως για παράδειγμα μία ρόδα. Η οποία δεν αντιμετωπίζει εμπόδια όταν πηγαίνει ευθεία αλλά αντιμετωπίζει τριβή όταν προσπαθεί να κινηθεί δεξιά και αριστερά.

Collision Bounds

Αφορά τα όρια που θέτουμε σε κάθε αντικείμενο σύμφωνα με το σχήμα του. Όταν του θέσουμε αυτά τα όρια τότε αυτός θα είναι και ο τρόπος που θα συμπεριφέρεται στο περιβάλλον(σαν κουτί, σφαίρα κτλ). Αν το αντικείμενο θέλουμε να μην έχει συγκεκριμένα όρια αλλά αποτελείται από πολλά σχήματα επιλέγουμε compound.

Create Obstacle

Για την δημιουργία εμποδίου και την αποφυγή του από τα υπόλοιπα.

ΚΕΦΑΛΑΙΟ 2

ΕΙΣΑΓΩΓΗ ΣΤΑ LOGIC BRICKS

2.1 ΕΙΣΑΓΩΓΗ ΣΤΟΥΣ ΑΙΣΘΗΤΗΡΕΣ(SENSORS)

Η πρώτη κατηγορία των τούβλων λογικής(Logic Bricks) που συναντάμε είναι οι Αισθητήρες(Sensors). Οι αισθητήρες αναγκάζουν τους ενεργοποιητές (actuators) να εκτελέσουν τις διεργασίες που τους θέσαμε. Συνδέονται μαζί τους μέσω των ελεγκτών(controllers). Δίνουν μια έξοδο όταν ένα γεγονός που ορίσαμε εμείς, πραγματοποιηθεί και στέλνουν έναν θετικό παλμό σε όλους τους συνδεδεμένους ελεγκτές. Βρίσκονται στην αριστερή πλευρά του πίνακα λογικής(Logic Panel).

Η στήλη ενός αισθητήρα αποτελείται από κουμπιά επιλογής του κατάλληλου τύπου αισθητήρα, καθώς και χώρο για το όνομα που θέλουμε να θέσουμε. Η στήλη των αισθητήρων περιλαμβάνει και μια επικεφαλίδα στην οποία ορίζεται ποιο επίπεδο λεπτομέρειας θα είναι ορατό στη στήλη. Με αυτόν τον τρόπο αποφεύγονται περιττοί αισθητήρες και λεπτομέρειες, ενώ οι χρήσιμοι χρησιμοποιούνται πιο εύκολα.

Οι τύποι των αισθητήρων είναι οι εξής:

Actuator : Ανιχνεύει πότε ένας συγκεκριμένος ενεργοποιητής λαμβάνει έναν σφυγμό ενεργοποίησης.

Always : Δίνει ένα συνεχές σήμα παραγωγής σε τακτά χρονικά διαστήματα.

Collision : Ανιχνεύει συγκρούσεις μεταξύ αντικειμένων ή υλικών.

Delay : Καθυστερεί την έξοδο με βάση ένα συγκεκριμένο αριθμό κρότων λογικής.

Joystick : Ανιχνεύει την κίνηση συγκεκριμένων κουμπιών ελέγχουν ενός joystick.

Keyboard : Ανιχνεύει την είσοδο από το πληκτρολόγιο.

Message : Ανιχνεύει είτε μηνύματα κειμένου είτε τιμές ιδιοτήτων.

Mouse : Ανιχνεύει γεγονότα που προέρχονται από το ποντίκι.

Near : Ανιχνεύει τα αντικείμενα που κινούνται σε συγκεκριμένη απόσταση από τον εαυτό τους.

Property : Ανιχνεύει αλλαγές στις ιδιότητες του αντικειμένου που ανήκει.

Radar : Ανιχνεύει τα αντικείμενα που κινούνται σε συγκεκριμένη απόσταση από τον εαυτό τους, μέσα σε μια γωνία από έναν άξονα.

Random : Παράγει τυχαίους παλμούς.

Ray : Πυροβολεί μια ακτίνα στην κατεύθυνση ενός άξονα και ανιχνεύει τα χτυπήματα.

Touch : Ανιχνεύει όταν ένα αντικείμενο είναι σε επαφή με ένα άλλο αντικείμενο.

2.1.1 ΚΟΙΝΕΣ ΕΠΙΛΟΓΕΣ

Υπάρχουν κάποιες επιλογές οι οποίες είναι κοινές σε όλους τους ελεγκτές.

Θα τις αναλύσουμε μία φορά στην αρχή και θα ισχύουν για κάθε έναν ξεχωριστά.

ΠΡΩΤΗ ΓΡΑΜΜΗ

Triangle Button(Τριγωνικό κουμπί): Όταν δεν θέλουμε οι πληροφορίες του αισθητήρα να είναι ορατές κυρίως λόγω χώρου, το κουμπί αυτό τις αναδιπλώνει σε ένα μία μόνο γραμμή με το όνομα του αισθητήρα ορατό, ώστε να μπορεί να εντοπίζεται.

Sensor type: Αναδυόμενο μενού από το οποίο μπορεί να αλλάξει ο τύπος του αισθητήρα.

Sensor name: Στο πεδίο αυτό υπάρχει αυτόματα ένα όνομα αισθητήρα, ανάλογο με τον τύπο του, το οποίο μπορεί να αλλάξει από τον χρήστη. Πρέπει να είναι μοναδικό, για να εντοπίζεται κυρίως όταν καλείται σε κώδικα Python.

X Button: Διαγράφει τον αισθητήρα.

Figure 14 : **2.1.1.1 Πρώτη γραμμή Κοινών Επιλογών**

ΔΕΥΤΕΡΗ ΓΡΑΜΜΗ

Ενεργοποίηση True Level: Εφόσον το κουμπί είναι ενεργοποιημένο, ο αισθητήρας θα συνεχίσει να στέλνει παλμούς TRUE στους ελεγκτές, για όσο η κατάστασή του θα είναι TRUE.

Ενεργοποίηση False Level: Εφόσον το κουμπί είναι ενεργοποιημένο, ο αισθητήρας θα συνεχίσει να στέλνει παλμούς FALSE στους ελεγκτές, για όσο η κατάστασή του θα είναι FALSE.

Freq: Η προεπιλεγμένη τιμή του πεδίου είναι 0. Όταν ενεργοποιήσουμε ένα από τα δύο προηγούμενα κουμπιά, τότε μπορούμε να επιλέξουμε την καθυστέρηση μεταξύ των επαναλαμβανόμενων παλμών. Η μέτρηση γίνεται σε λογικούς κρότους οι οποίοι έχουν μια συχνότητα των 60 τικ το δευτερόλεπτο.

Level: Δίνει σήμα να ενεργοποιηθούν οι Ελεγκτές όταν η κατάσταση αλλάζει σε νέα αξία.

Tap: Ακόμα και όταν ο αισθητήρας είναι σε κατάσταση TRUE, η επιλογή αυτή θα αλλάξει την κατάσταση του σε FALSE μέσα στο επόμενο frame.

Invert: Όταν είναι επιλεγμένο το συγκεκριμένο κουμπί τότε η κατάσταση του αισθητήρα θα αλλάζει. Θα στέλνονται δηλαδή TRUE παλμοί, εκεί που πρέπει να στέλνονται FALSE και FALSE όταν θα έπρεπε να στέλνονται TRUE.

Figure 15 : 2.1.1.2 Δεύτερη γραμμή Κοινών Επιλογών

2.1.2 ΑΙΣΘΗΤΗΡΕΣ

Actuator

Ο αισθητήρας αυτός αφορά τους ενεργοποιητές. Εντοπίζει τον ενεργοποιητή που θέσαμε και ανιχνεύει το πότε ενεργοποιείται. Όταν συμβεί αυτό στέλνει έναν παλμό TRUE ώστε να συμβούν οι κατάλληλες διεργασίες. Όταν ο ενεργοποιητής απενεργοποιηθεί στέλνει έναν σφυγμό FALSE.

Ειδική Επιλογή:

→Name of Actuator: το όνομα του ενεργοποιητή που θέλουμε να ελέγχεται.

Figure 16 : **2.1.2.1 Actuator Sensor**

Always

Ο αισθητήρας αυτός χρησιμοποιείται για διεργασίες που πρέπει να συμβούν σε κάθε λογικό κτύπο. Όταν ενεργοποιείται στέλνει έναν θετικό παλμό στον ελεγκτή με τον οποίο είναι συνδεδεμένος.

Δεν έχει ειδικές επιλογές.

Figure 17 : **2.1.2.2 Always Sensor**

Collision

Ο αισθητήρας αυτός ανιχνεύει όλες τις συγκρούσεις με μια ιδιότητα ή με ένα υλικό που ορίσαμε. Μόνο ιδιότητες και υλικά με το όνομα που θέσαμε θα παράγουν ένα θετικό παλμό προς τον ελεγκτή. Αν το φίλτρο παραμείνει κενό τότε θα υπολογιστεί η σύγκρουση που θα συμβεί με το πρώτο αντικείμενο που θα βρεθεί.

Ειδικές Επιλογές:

→Pulse Button: Το αντικείμενο είναι αισθητό και σε άλλες συγκρούσεις ακόμη και αν βρίσκεται ακόμη σε επαφή με το αντικείμενο το οποίο προκάλεσε θετικό παλμό.

→M/P Button: Πραγματοποιεί εναλλαγή μεταξύ υλικού και ιδιότητας για το πεδίο φιλτραρίσματος.

Figure 18 : **2.1.2.3 Collision Sensor**

Delay

Ο αισθητήρας αυτός δημιουργήθηκε για να καθυστερεί κάποια γεγονότα να συμβούν για ένα συγκεκριμένο χρονικό διάστημα (λογικά τικ), ορισμένο από τον χρήστη.

Ειδικές Επιλογές:

→Delay

Ο αισθητήρας θα περιμένει τόσα λογικά τικ, όσα ορίσαμε, πριν στείλει θετικό παλμό στον ελεγκτή.

→Duration

Ο αισθητήρας θα περιμένει τόσα λογικά τικ, όσα ορίσαμε, πριν στείλει αρνητικό παλμό στον ελεγκτή.

→Repeat Button

Όταν τελειώσει ο χρόνος των πεδίων Delay και Duration, ο αισθητήρας θα ξεκινήσει από την αρχή, εφόσον το συγκεκριμένο πεδίο είναι επιλεγμένο.

Figure 19 : **2.1.2.4 Delay Sensor**

Joystick

Ο αισθητήρας Joystick αναγνωρίζει το συνδεδεμένο με τον υπολογιστή χειριστήριο παιχνιδιών. Ενεργοποιείται όταν αυτό εκτελεί μια κίνηση. Ανιχνεύει τόσο τα γεγονότα, όσο και τα κουμπιά και άλλους ελέγχους που διαθέτει. Μπορούν να χρησιμοποιηθούν παραπάνω από ένα χειριστήρια.

Ειδικές Επιλογές:

→Index

Ορίζει ποιο από τα συνδεδεμένα χειριστήρια θα χρησιμοποιηθεί.

→Event Type

Στο αναδυόμενο μενού εντοπίζουμε τέσσερις τύπους γεγονότων. Κάθε φορά επιλέγουμε αυτό που θέλουμε να χρησιμοποιηθεί.

Single Axis

Ανιχνεύει κίνηση σε έναν άξονα του χειριστηρίου.

Axis Number

1 = Οριζόντιος άξονας (αριστερά/ δεξιά)

2 = Κάθετος άξονας (μπροστά/ πίσω)

3 = Άξονας κουμπιών (πάνω/ κάτω)

4 = Joystick άξονα περιστροφής (αριστερά/ δεξιά)

Axis Threshold

Κατώτατο όριο για το οποίο το Joystick ενεργοποιείται(Ακτίνα 0 - 32768)

Hat

Ανίχνευση κίνησης ενός συγκεκριμένου hat ελέγχου του joystick.

Hat number

Διευκρινίζει ποιο hat θα χρησιμοποιηθεί.

Hat Direction

Διευκρινίζει την κατεύθυνση που θα χρησιμοποιηθεί: πάνω, κάτω, αριστερά, δεξιά ή και συνδυασμένα.

Axis

Axis Number

Προσδιορίζει τον άξονα (1 ή 2)

Axis Threshold

Κατώτατο όριο στο οποίο το Joystick ενεργοποιείται (Ακτίνα 0 - 32768)

Axis Direction

Διευκρινίζει την κατεύθυνση που θα χρησιμοποιηθεί.

Button

Προσδιορίζει το button number που έχει χρησιμοποιηθεί από τον χρήστη.

→All Events

Ο Αισθητήρας ενεργοποιείται για όλα τα γεγονότα στον τρέχοντα τύπο αυτού του πηδαλίου.

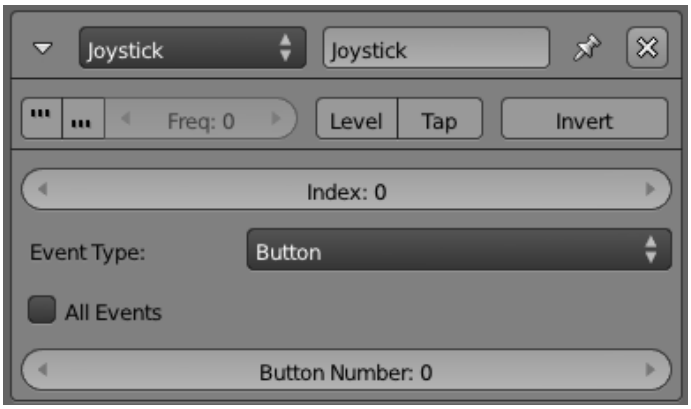

Figure 20 : **2.1.2.5 Joystick Sensor**

Keyboard

Ο αισθητήρας αυτός αναγνωρίζει την είσοδο από το πληκτρολόγιο και μπορεί να την αποθηκεύσει σε μια String Property. Μόλις την ανιχνεύσει στέλνει έναν θετικό παλμό στον συνδεδεμένο ελεγκτή-ές.

Ειδικές Επιλογές:

→Key: Στο πεδίο αυτό ορίζουμε ένα κλειδί. Κάνουμε κλικ στο κενό και εισάγουμε ένα κλειδί από το πληκτρολόγιο. Αυτό είναι και το ενεργό κλειδί που θα ενεργοποιήσει έναν παλμό TRUE. Μόλις το κλειδί απενεργοποιηθεί θα σταλθεί ένας παλμός FALSE.

→All keys Button: Όταν πατηθεί οποιοδήποτε πλήκτρο από το πληκτρολόγιο θα ενεργοποιηθεί ο παλμός TRUE.

→First Modifier

→Second Modifier: Τα δύο αυτά πεδία δέχονται κλειδιά από το πληκτρολόγιο. Χρησιμοποιούνται σε περίπτωση που θέλουμε ο αισθητήρας να ενεργοποιηθεί όταν εκτός από το βασικό κλειδί είναι και αυτά πατημένα. Μόνο τότε στέλνεται θετικός παλμός. Πολύ χρήσιμα για συνδυασμούς πλήκτρων όπως π.χ. Ctrl+Alt+Delete

→LogToggle: Στο πεδίο αυτό μπορεί να γίνει μια εκχώρηση μιας Boolean μεταβλητής, η οποία όταν είναι TRUE θα ορίζει αν θα γίνεται καταγραφή των πλήκτρων σε ένα String.

→Target: Στο πεδίο εισάγουμε το όνομα μιας ιδιότητας (property) που πρέπει να είναι τύπου String. Σε αυτήν θα αποθηκευτούν τα πλήκτρα που πληκτρολογήσαμε και θέσαμε ως κλειδιά εφόσον το επιθυμούμε.

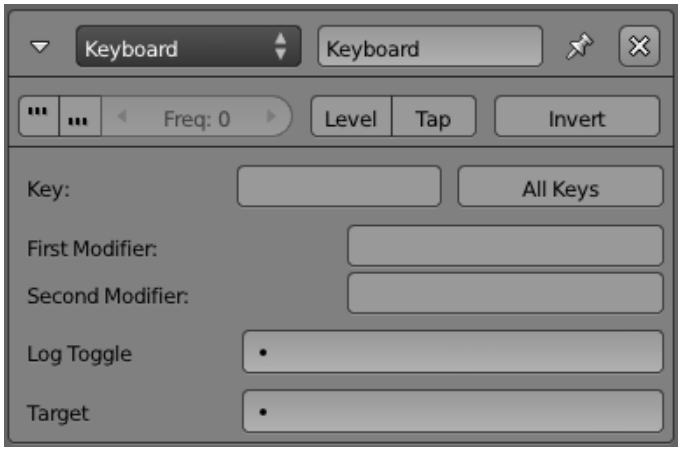

Figure 21 : **2.1.2.6 Keyboard Sensor**

Message

Ο αισθητήρας αυτός δέχεται και ανιχνεύει μηνύματα κειμένου με συγκεκριμένο ή όχι θέμα. Ρυθμίζεται έτσι ώστε να στέλνει έναν θετικό παλμό μόλις το μήνυμα σταλθεί από οπουδήποτε στη μηχανή. Ανιχνεύει και τιμές ιδιοτήτων.

Ειδικές Επιλογές:

→Subject: Αν θέλουμε το μήνυμα που θα ληφθεί να έχει κατάλληλο και συγκεκριμένο θέμα συμπληρώνουμε το πεδίο. Αλλιώς μπορεί να αφεθεί και κενό.

Figure 22 : **2.1.2.7 Message Sensor**

Mouse

Οποιοδήποτε γεγονός έχει να κάνει με την κίνηση του ποντικιού, ανιχνεύεται από αυτόν τον αισθητήρα.

Ειδικές Επιλογές:

Εδώ συναντάμε μια λίστα με τους διάφορους τύπους των γεγονότων που μπορεί να προέλθουν από το ποντίκι.

→Mouse over any: Μόλις το ποντίκι κινηθεί πάνω από οποιοδήποτε αντικείμενο, ένας παλμός TRUE στέλνεται.

→Mouse over: Μόλις το ποντίκι κινηθεί πάνω από το αντικείμενο που είναι ιδιοκτήτης του αισθητήρα, ένας παλμός TRUE στέλνεται.

→Movement: Μόλις το ποντίκι κινηθεί κατά οποιονδήποτε τρόπο, προκαλείται ένα ρεύμα από παλμούς TRUE.

→Wheel Down: Όταν ο τροχός του ποντικιού μετακινείται προς τα κάτω, προκαλείται ένα ρεύμα από παλμούς TRUE.

→Wheel Up: Όταν ο τροχός του ποντικιού μετακινείται προς τα πάνω, προκαλείται ένα ρεύμα από παλμούς TRUE.

→Right button: Όταν γίνεται δεξί κλικ, ένας παλμός TRUE στέλνεται.

→Left button: Όταν γίνεται αριστερό κλικ, ένας παλμός TRUE στέλνεται.

→Middle button: Όταν γίνεται κλικ στον τροχό, ένας παλμός TRUE στέλνεται.

Μόλις οποιαδήποτε από τις παραπάνω συνθήκες τελειώσει, στέλνεται ένας παλμός FALSE.

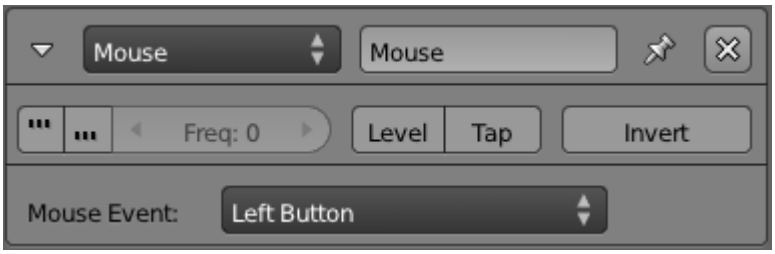

Figure 23 : **2.1.2.8 Mouse Sensor**

Near

Ο αισθητήρας αυτός εντοπίζει τα αντικείμενα που κινούνται σε συγκεκριμένη απόσταση, την οποία έχουμε θέσει, από τον εαυτό τους. Για να γίνει ο εντοπισμός τα αντικείμενα πρέπει να έχουν ενεργοποιημένη την ιδιότητα “Actor”.

Ειδικές Επιλογές:

→Property: Σε αυτό το πεδίο θέτουμε κάποια ιδιότητα την οποία έχει ένα ή περισσότερα αντικείμενα, τα οποία επιθυμούμε να ανιχνεύει ο αισθητήρας μας.

→Distance: Μετριέται σε μονάδες Blender και αφορά την απόσταση για την οποία θα ανιχνεύονται αντικείμενα.

→Reset: Αφορά και πάλι την απόσταση και ορίζει το πόση πρέπει να είναι ώστε ο αισθητήρας να επανέρχεται, δηλαδή να στέλνει παλμό FALSE.

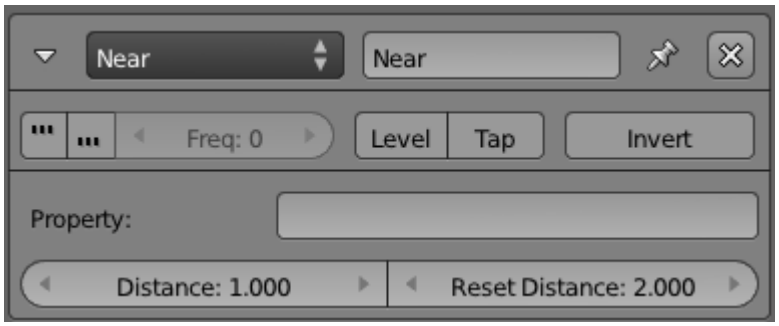

Figure 24 : **2.1.2.9 Near Sensor**

Property

Όταν συμβαίνουν αλλαγές στις ιδιότητες του αντικειμένου που διακατέχει τον συγκεκριμένο αισθητήρα, τότε ο Property τις εντοπίζει και στέλνει έναν θετικό παλμό.

Ειδικές Επιλογές:

→Evaluation Type: Περιέχει τέσσερις διαφορετικούς τρόπους με τους οποίους θα γίνεται η αξιολόγηση της τιμής.

Changed: Μόλις αλλάξει η τιμή της ιδιότητας τότε στέλνει έναν θετικό παλμό.

Interval: Έχουμε δύο νέα πεδία στα οποία θέτουμε μικρότερη και μεγαλύτερη τιμή. Ελέγχεται αν η τιμή της ιδιότητας βρίσκεται ανάμεσα σε αυτά τα όρια. Τότε μόνο στέλνεται ένας θετικός παλμός.

Not Equal: Όταν η τιμή που έχει η ιδιότητα είναι διαφορετική από αυτήν που θέσαμε στον αισθητήρα, τότε στέλνεται ο θετικός παλμός.

Equal: Όταν η τιμή που έχει η ιδιότητα είναι ίση από αυτήν που θέσαμε στον αισθητήρα, τότε στέλνεται ο θετικός παλμός.

→Property: Εισάγουμε το όνομα της ιδιότητας που θέλουμε να ελέγξουμε.

→Value: Εισάγουμε την τιμή της ιδιότητας για την οποία θα γίνονται η αλλαγές σύμφωνα με τον Evaluation Type.

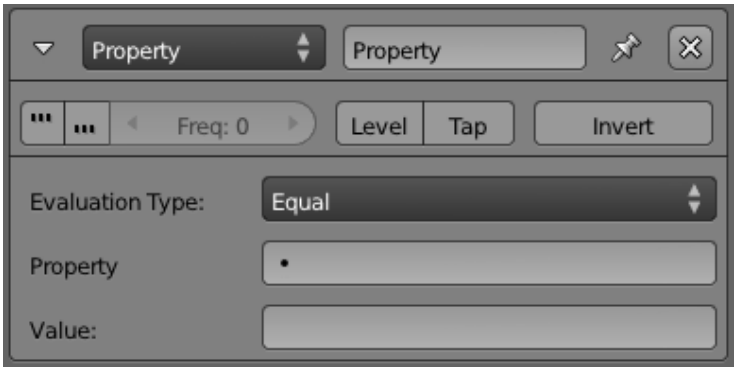

Figure 25 : **2.1.2.10 Property Sensor**

Radar

Ο αισθητήρας αυτός λειτουργεί παρόμοια με τον αισθητήρα Near. Η βασική διαφορά είναι πως εντοπίζει αντικείμενα μέσα σε μια ακτίνα η οποία σχηματίζεται ως γωνία από έναν άξονα. Το κέντρο των αντικειμένων είναι η κορυφή και η απόσταση από έναν άξονα είναι η βάση, έτσι ώστε να σχηματίζεται ένας αόρατος κώνος. Μια παρατήρηση και σε αυτόν τον αισθητήρα είναι πως τα αντικείμενα πρέπει να έχουν το "Actor" ενεργό για να ανιχνευθούν.

Ειδικές Επιλογές:

→Property: Στο πεδίο αυτό μπορούμε να εισάγουμε μία ιδιότητα, έτσι ώστε ο αισθητήρας να ψάχνει αντικείμενα που να την κατέχουν.

→Axis: Καθορίζεται η κατεύθυνση του αόρατου κώνου.

→Angle: Καθορίζεται η γωνία του κώνου (Ακτίνα: 0.00 μέχρι 179.9 βαθμούς).

→Distance: Μετριέται σε μονάδες Blender και καθορίζει το μήκος του κώνου.

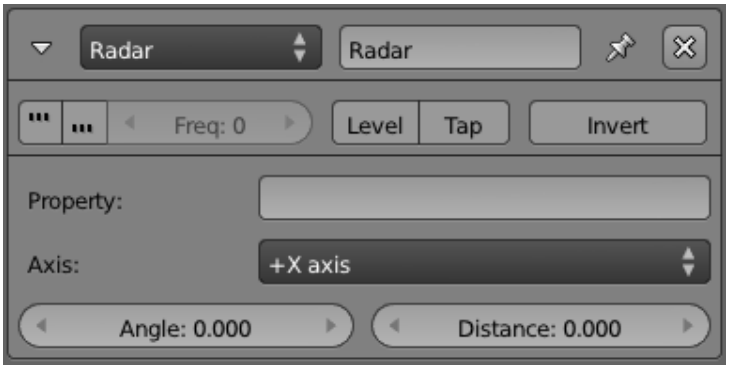

Figure 26 : **2.1.2.11 Radar Sensor**

Random

Ο αισθητήρας αυτός παράγει τυχαίους παλμούς με έναν ρυθμό που θέτουμε εμείς.

Ειδικές Επιλογές:

→Literal|Seed: Ο αισθητήρας random λειτουργεί βάση ενός αλγορίθμου παραγωγής τυχαίων αριθμών. Δίνοντας μια τιμή στο πεδίο αυτό καθορίζεται από ποιον αριθμό θα αρχίσει αυτός ο αλγόριθμος (Ακτίνα 0-1000).

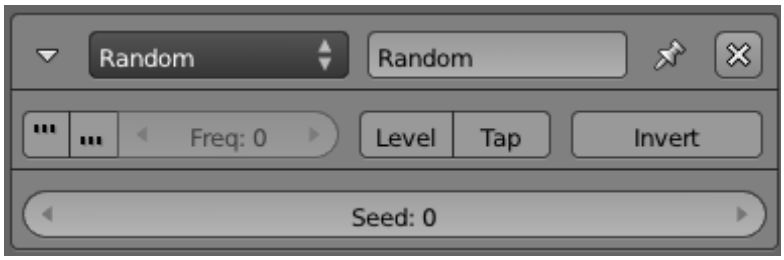

Figure 27 : **2.1.2.12 Random Sensor**

Ray

Ο αισθητήρας αυτός στέλνει έναν παλμό TRUE μόλις χτυπήσει κάτι. Για να το πετύχει αυτό πυροβολεί σε μια ακτίνα στην κατεύθυνση ενός από τους άξονες. Έχει τη δυνατότητα να δεχτεί φίλτρα σύμφωνα με τα οποία θα εντοπίζει αντικείμενα με συγκεκριμένη ιδιότητα ή υλικό. Τα αντικείμενα πρέπει να έχουν το "Actor" ενεργό για να ανιχνευθούν.

Ειδικές Επιλογές:

→Property: Εναλλάσσεται με την επιλογή Material και περιορίζει τον αισθητήρα να ψάξει για αντικείμενα με τη συγκεκριμένη ιδιότητα ή υλικό, ανάλογα με την επιλογή μας.

→Axis: Καθορίζει την κατεύθυνση της ακτίνας.

→Range: Καθορίζει το μήκος της ακτίνας.

→X-Ray Mode button: Χρησιμοποιεί το πεδίο Property-Material. Με ακτίνες x-ray διαπερνά την ιδιότητα ή υλικό που θέσαμε στο αντικείμενό μας.

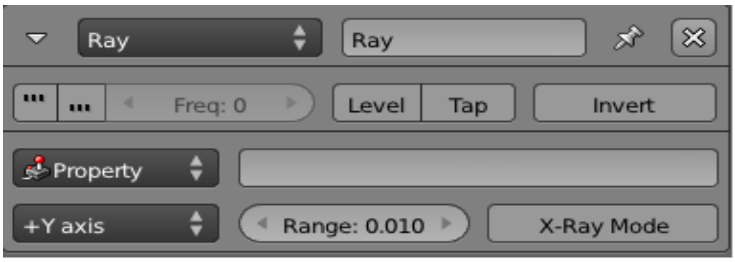

Figure 28 : **2.1.2.13 Ray Sensor**

Touch

Όταν το αντικείμενό μας βρίσκεται σε επαφή με ένα άλλο αντικείμενο , τότε ο αισθητήρας αυτός στέλνει έναν θετικό παλμό. Ένας παλμός TRUE θα σταλθεί κατά την επαφή και ένας FALSE μόλις σταματήσει.

Ειδικές Επιλογές:

→Material: Αν θέσουμε ένα υλικό σε αυτό το πεδίο τότε μόνο η επαφή μαζί του θα προκαλέσει έναν θετικό παλμό. Αν αφεθεί κενό, ο παλμός θα σταλθεί κατά την επαφή με οποιοδήποτε αντικείμενο.

Figure 29 : **2.1.2.14 Touch Sensor**

2.2 ΕΙΣΑΓΩΓΗ ΣΤΟΥΣ ΕΛΕΓΚΤΕΣ(CONTROLLERS)

Οι ελεγκτές είναι τα Logic Bricks που ενώνουν τους αισθητήρες με τους ενεργοποιητές. Οι αισθητήρες στέλνουν τις πληροφορίες στους ελεγκτές και αφού εκτελεστούν οι διαδικασίες λογικής, στέλνονται οι θετικοί παλμοί για την ενεργοποίηση των ενεργοποιητών. Στην πραγματικότητα η δουλειά των ελεγκτών είναι να ελέγχουν τους παλμούς που προκαλούν οι αισθητήρες(θετικούς ή αρνητικούς), να τους συνδυάζουν και να στέλνουν την κατάλληλη απάντηση στους ενεργοποιητές.

Τύποι Ελεγκτών:

- AND Ελεγκτής
- OR Ελεγκτής
- NAND Ελεγκτής
- NOR Ελεγκτής

- XOR Ελεγκτής
- XNOR Ελεγκτής
- Expression Ελεγκτής
- Python Ελεγκτής

2.2.1 ΚΟΙΝΕΣ ΕΠΙΛΟΓΕΣ

Υπάρχουν κάποιες επιλογές οι οποίες είναι κοινές σε όλους τους ελεγκτές.

Θα τις αναλύσουμε μία φορά στην αρχή και θα ισχύουν για κάθε έναν ξεχωριστά.

Controller Type: Είναι ένα αναδυόμενο μενού στο οποίο γίνεται η επιλογή ο τύπος του ελεγκτή.

Controller Name: Σε αυτό το πεδίο εισάγεται το όνομα του ελεγκτή, το οποίο είναι μοναδικό.

State Index: Δείχνει σε ποια από τις καταστάσεις ο ελεγκτής θα είναι ενεργός και θα λειτουργήσει.

Preference Button: Αν θέσουμε το κουμπί αυτό on τότε αυτός ο ελεγκτής θα προηγηθεί των υπολοίπων.

X Button: Διαγράφει τον αισθητήρα.

2.2.2 ΕΛΕΓΚΤΕΣ:

AND

Μια θετική έξοδος TRUE δίνεται όταν:

Όλες οι είσοδοι είναι TRUE και το αντικείμενο βρίσκεται σε κατάσταση σχεδίασης.

Διαφορετικά δίνει έξοδο FALSE.

Figure 30 : **2.2.2.1 And Controller**

OR

Μια θετική έξοδος TRUE δίνεται όταν:

Μία ή περισσότερες από τις εισόδους είναι TRUE και το αντικείμενο βρίσκεται σε κατάσταση σχεδίασης.

Διαφορετικά δίνει έξοδο FALSE.

Figure 31 : **2.2.2.2 Or Controller**

NAND

Αυτός ο Ελεγκτής ενεργοποιεί όλους τους συνδεδεμένους Ενεργοποιητές αν

- το αντικείμενο του παιχνιδιού βρίσκεται σε ορισμένη κατάσταση
- τουλάχιστον ένας συνδεδεμένος αισθητήρας ενεργοποιεί τον ελεγκτή
- τουλάχιστον ένας συνδεδεμένος αισθητήρας αξιολογείται False

Αυτός ο Ελεγκτής απενεργοποιεί όλους τους συνδεδεμένους Ενεργοποιητές αν

- το αντικείμενο του παιχνιδιού βρίσκεται σε ορισμένη κατάσταση
- τουλάχιστον ένας συνδεδεμένος αισθητήρας ενεργοποιεί τον ελεγκτή
- ΟΛΟΙ οι συνδεδεμένοι αισθητήρες αξιολογούνται True

Figure 32 : **2.2.2.3 Nand Controller**

NOR

Μια θετική έξοδος TRUE δίνεται όταν:

Καμιά από τις εισόδους δεν είναι TRUE και το αντικείμενο βρίσκεται σε κατάσταση σχεδίασης.

Διαφορετικά δίνει έξοδο FALSE.

Figure 33: **2.2.2.4 Nor Controller**

XOR

Μια θετική έξοδος TRUE δίνεται όταν:

Μία μόνο από τις εισόδους είναι TRUE και το αντικείμενο βρίσκεται σε κατάσταση σχεδίασης.

Διαφορετικά δίνει έξοδο FALSE.

Figure 34 : **2.2.2.5 Xor Controller**

XNOR

Μια θετική έξοδος TRUE δίνεται όταν:

Μία μόνο από τις εισόδους είναι FALSE και το αντικείμενο βρίσκεται σε κατάσταση σχεδίασης.

Διαφορετικά δίνει έξοδο FALSE.

Figure 35 : **2.2.2.6 Xnor Controller**

Expression

Σε αυτόν τον ελεγκτή ο χρήστης γράφει μια έκφραση και μια θετική έξοδος TRUE δίνεται όταν:

Το αποτέλεσμα της εξόδου είναι TRUE και το αντικείμενο βρίσκεται σε κατάσταση σχεδίασης.

Διαφορετικά δίνει έξοδο FALSE.

Οι εκφράσεις που χρησιμοποιούνται, αποτελούνται από μεταβλητές, σταθερές και χειριστές.

→Μεταβλητές: τα ονόματα των αισθητήρων

Οι ιδιότητες που θέτει ο χρήστης σε αντικείμενα

→Χειριστές: Λογικοί→ *, /, +, -

Σύγκρισης→ <, >, >=, <=, ==, !=

Boolean→ AND, OR, NOT

→Λειτουργίες: Λογικές, μαθηματικές

Figure 36 : **2.2.2.7 Expression Controller**

Python

Ο ελεγκτής Python ελέγχει την είσοδο χρησιμοποιώντας ένα script προγραμματισμένο από τον ίδιο τον χρήστη.

Έχει δύο λειτουργίες:

Python: Στην επιλογή αυτή μπορούμε στο κενό πεδίο να εισάγουμε το όνομα του script που θα χρησιμοποιήσουμε. Παρατηρούμε πως με ένα κλικ στο πεδίο αυτό εμφανίζεται η λίστα με τα ήδη έτοιμα προγράμματα.

Module: Στην επιλογή αυτή εισάγουμε το όνομα του script που θα χρησιμοποιηθεί και έπειτα με την τελεία (.) καλούμε την function που θέλουμε να εκτελεστεί και φυσικά ανήκει στο συγκεκριμένο πρόγραμμα.

Και οι δύο μπορούν να γραφτούν στον Text Editor και να αποθηκευτούν μέσα στα .blend αρχεία, ή να παραμείνουν ως εξωτερικά script.

Figure 37 : **2.2.2.8 Python Controller**

2.3 ΕΙΣΑΓΩΓΗ ΣΤΟΥΣ ΕΝΕΡΓΟΠΟΙΗΤΕΣ(ACTUATORS)

Οι ενεργοποιητές είναι συνδεδεμένοι με τους ελεγκτές και περιμένουν έναν θετικό παλμό. Μόλις τον λάβουν είναι υπεύθυνοι για τη δημιουργία κίνησης, την αναπαραγωγή ήχου, την δημιουργία αντικειμένων, την εναλλαγή μεταξύ σκηνών, τον τερματισμό μιας σκηνής και άλλα.

Τύποι ενεργοποιητών:

- Action
- Camera
- Constraint
- Edit Object
- Filter 2D
- Game

- Message
- Motion
- Parent
- Property
- Random
- Scene
- Sound
- State
- Steering
- Visibility

2.3.1 ΚΟΙΝΕΣ ΕΠΙΛΟΓΕΣ:

Υπάρχουν κάποιες επιλογές οι οποίες είναι κοινές σε όλους τους ενεργοποιητές.

Θα τους αναλύσουμε μία φορά στην αρχή και θα ισχύουν για κάθε έναν ξεχωριστά.

Triangle button: Το μικρό αυτό τριγωνικό κουμπί μαζεύει όλες τις πληροφορίες του ενεργοποιητή σε μια μόνο γραμμή για εξοικονόμηση χώρου.

Actuator type: Μας διευκρινίζει τον τύπο του ενεργοποιητή που επιλέξαμε.

Actuator name: Στο κενό αυτό μπορούμε να εισάγουμε το όνομα που θέλουμε να δώσουμε στον ενεργοποιητή. Το όνομα αυτό πρέπει να είναι μοναδικό ανάμεσα στα επιλεγμένα αντικείμενα.

X Button: Το κουμπί που διαγράφει τον ενεργοποιητή.

2.3.2 ΕΝΕΡΓΟΠΟΙΗΤΕΣ:

Action

Ο ενεργοποιητής Action μπορεί μόνο να δημιουργηθεί σε αντικείμενα armature. Χρησιμοποιείται για να αποδώσει κινήσεις σε αυτά. Ταυτόχρονα ρυθμίζει και τον τρόπο αναπαραγωγής τους.

Ειδικές Επιλογές:

→Action Playback Type

Play: Μόλις ενεργοποιηθεί με έναν παλμό TRUE θα αρχίσει να παίζει από την αρχή μέχρι τέλος για μία φορά.

Flipper: Αν ληφθεί παλμός TRUE θα παίξει μία φορά από την αρχή μέχρι τα τέλος. Αν ληφθεί παλμός FALSE τότε θα παίξει αντίστροφα από το δεδομένο frame ως την αρχή.

Loop End: Μόλις ενεργοποιηθεί θα παίζει συνεχόμενα από το τέλος μέχρι την αρχή έως ότου ληφθεί ένας αρνητικός παλμός. Όταν ο παλμός αυτός ληφθεί θα σταματήσει μόνο όταν φτάσει το τέλος της επανάληψης.

Loop Start: Μόλις ενεργοποιηθεί θα παίζει συνεχόμενα από την αρχή μέχρι το τέλος έως ότου ληφθεί ένας αρνητικός παλμός. Όταν ο παλμός αυτός ληφθεί θα σταματήσει μόνο όταν φτάσει το τέλος της επανάληψης.

Property: Η κίνηση θα παίξει μόνο από το frame που ορίστηκε στο πεδίο αυτό.

→ Action

Επιλέγει τη δράση που θα χρησιμοποιηθεί.

→ Continue

Αποκαθιστά το τελευταίο πλαίσιο κατά την ενεργοποίηση/απενεργοποίηση, διαφορετικά παίζει από την αρχή κάθε φορά.

→Start Frame

Θέτει το αρχικό πλαίσιο της δράσης

→End Frame

Θέτει το τελικό πλαίσιο της δράσης

→Blendin

Αριθμός πλαισίων του συνδυασμών κινήσεων.

→Priority

Προτεραιότητα εκτέλεσης - οι χαμηλότεροι αριθμοί θα αγνοήσουν τις ενέργειες με τους υψηλότερους αριθμούς.

→Frame Property

Εκχωρεί τον τρέχοντα αριθμό πλαισίου της δράσης σε αυτή την ιδιότητα.

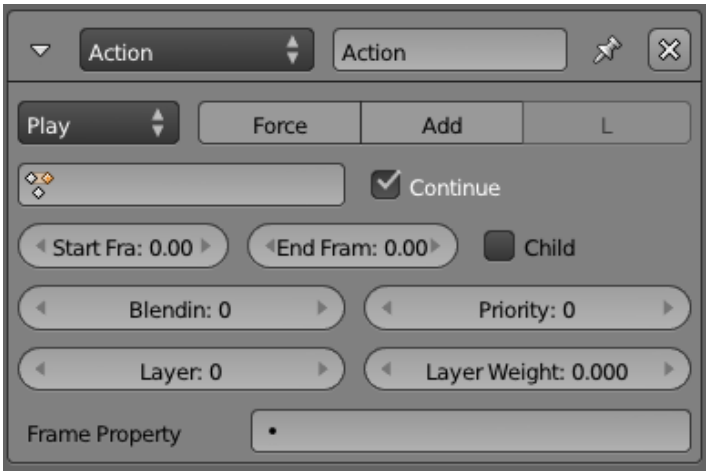

Figure 38 : **2.3.2.1 Action Actuator**

Camera

Ο ενεργοποιητής αυτός προκαλεί την κάμερα να ακολουθεί ή να εντοπίζει ένα συγκεκριμένο αντικείμενο.

Ειδικές Επιλογές:

→Camera Object

Το αντικείμενο το οποίο ακολουθεί ή εντοπίζει η κάμερα.

→Height

Το ύψος στο οποίο προσπαθεί να μείνει η κάμερα πάνω από το κέντρο του αντικειμένου του παιχνιδιού.

→Axis

Ο άξονας τον οποίο ακολουθεί η κάμερα (X ή Y)

→Min

Η ελάχιστη απόσταση από την οποία η κάμερα θα ακολουθεί το αντικείμενο του παιχνιδιού.

→Max

Η μέγιστη απόσταση από την οποία η κάμερα θα ακολουθεί το αντικείμενο του παιχνιδιού.

→Damping

Η δύναμη του περιορισμού που οδηγεί τη κάμερα πίσω από το στόχο. Εμβέλεια: 0 μέχρι 10. Όσο ψηλότερη η παράμετρος, τόσο πιο γρήγορα η κάμερα θα προσαρμόζεται για να βρίσκεται μέσα στο περιορισμένο εύρος.

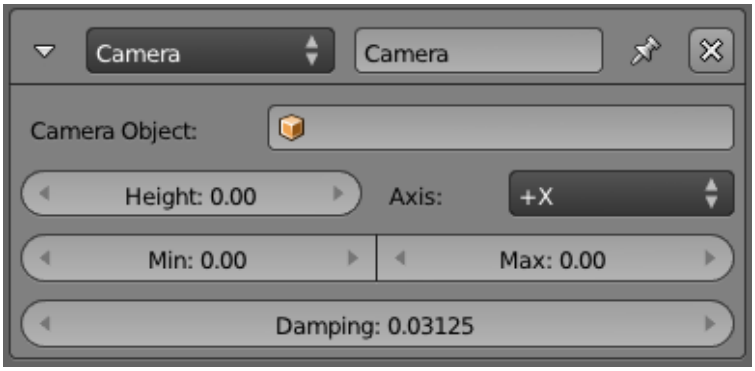

Figure 39 : **2.3.2.2 Camera Actuator**

Constraint

Ο ενεργοποιητής αυτός αφορά την θέση και τον προσανατολισμό του αντικειμένου. Δουλειά του είναι να θέτει έναν περιορισμό.

Ειδικές Επιλογές:

→Constraint Mode

Εντοπίζουμε τέσσερις τύπους περιορισμού:

Force Field Constraint

Δημιουργεί ένα δυναμικό πεδίο ουδέτερης ζώνης κατά μήκος ενός από τους άξονες του αντικειμένου.

Orientation Constraint

Περιορίζει το διευκρινισμένο άξονα στο παιχνίδι σε μια συγκεκριμένη κατεύθυνση ως προς τον παγκόσμιο άξονα.

Distance Constraint

Διατηρεί την απόσταση την οποία πρέπει να έχει το αντικείμενο του παιχνιδιού από την επιφάνεια.

Location Constraint

Περιορίζει τη θέση του αντικειμένου του παιχνιδιού μέσα στην κατεύθυνση του παγκόσμιου άξονα.

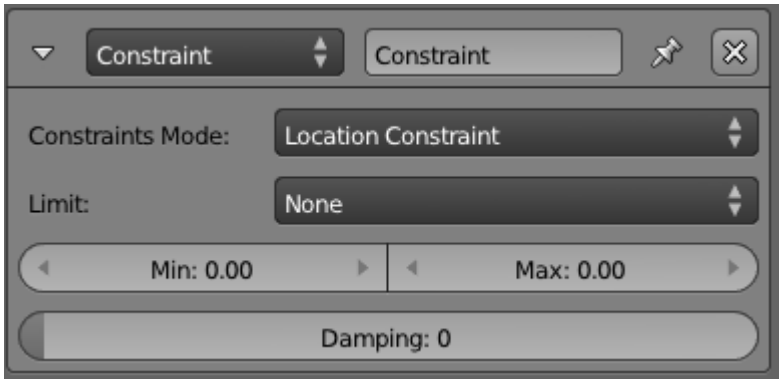

Figure 40 : **2.3.2.3 Constraint Actuator**

Edit Object

Ο ενεργοποιητής αυτός επιτρέπει στον χρήστη να ρυθμίσει ένα αντικείμενο στο παιχνίδι, μέσα από τις επιλογές που του παρουσιάζονται.

Ειδικές Επιλογές:

→Edit Object

Το πεδίο αυτό αποτελείται από κάποιες επιλογές:

Dynamics:

Εμφανίζεται ένα μενού από δυναμικές λειτουργίες τις οποίες μπορεί να θέσει ο χρήστης στο αντικείμενο.

Dynamic Operations:

- Set Mass

  Επιτρέπει στον χρήστη να θέσει τη μάζα του τρέχοντος αντικειμένου για τα Physics(Ακτίνα 0 - 10,000).

- Disable Rigid Body

  Απενεργοποιεί την κατάσταση Rigid Body από το αντικείμενο - απενεργοποιεί τη σύγκρουση.

- Enable Rigid Body

  Ενεργοποιεί την κατάσταση Rigid Body στο αντικείμενο - ενεργοποιεί την σύγκρουση.

- Suspend Dynamics

  Αναστέλλει τα dynamics του αντικειμένου (ταχύτητα του αντικειμένου).

- Restore Dynamics

  Επαναφέρει τα dynamics του αντικειμένου (ταχύτητα του αντικειμένου).

Track To:

Αναγκάζει ένα αντικείμενο να «κοιτάει» ένα άλλο αντικείμενο σε 2D ή 3D. Θεωρεί τον άξονα Y μπροστά από το αντικείμενο.

- Object

  Το αντικείμενο που ακολουθεί.

- Time

  Ο αριθμός των πλαισίων που θα χρειαστεί για να στραφεί προς το αντικείμενο-στόχο (Ακτίνα 0-2000).

- 3D Button(toggle).

  Ενεργοποιεί την παρακολούθηση 2D (X,Y) ή 3D (X,Y,Z).

Replace Mesh

Αντικαθιστά το τρέχον αντικείμενο με ένα άλλο. Και το αντικείμενο και η φυσική του μπορούν να αντικατασταθούν, μαζί ή ανεξάρτητα.

- Mesh

  το όνομα του αντικειμένου που αντικαθιστά το τρέχον αντικείμενο.

- Gfx Button

  αντικαθιστά το ορατό αντικείμενο.

- Phys Button

  αντικαθιστά τα physics του αντικειμένου (μη σύνθετες μορφές)

End Object

Καταστρέφει το τρέχον αντικείμενο

Add Object

Στο κέντρο του τρέχοντος αντικειμένου προστίθεται ένα άλλο. Το νέο αυτό αντικείμενο πρέπει να βρίσκεται σε ένα άλλο στρώμα(layer).

- Object

  Στο πεδίο αυτό βάζουμε το όνομα του αντικειμένου που θα προστεθεί.

- Time

  Προσθέτουμε τον χρόνο(σε καρέ) για τον οποίο το αντικείμενο παραμένει ζωντανό. Όταν τελειώσει ο χρόνος το αντικείμενο καταστρέφεται. Αν είναι μηδέν τότε το αντικείμενο θα παραμείνει για πάντα.

- Linear Velocity

  Η γραμμική ταχύτητα δημιουργεί μια κίνηση στο δημιουργημένο αντικείμενο. Πολύ χρήσιμο στο να δημιουργεί αντικείμενα με αρχική ταχύτητα τα οποία θα πυροβολούν.

- Angular Velocity

  Προσθέτει γωνιακή ταχύτητα στο δημιουργημένο αντικείμενο.

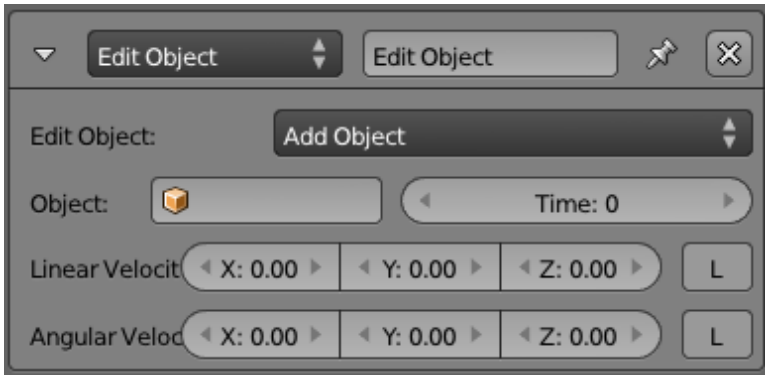

Figure 41 : **2.3.2.4 Edit Object Actuator**

2D Filter

Τα φίλτρα χρησιμοποιούνται στις εικόνες και τις φιλτράρουν ανάλογα με την απόδοση που θέλουμε να δώσουμε. Εφαρμόζονται λίγο πριν αποδοθεί τελικά ένα αντικείμενο.

Υπάρχουν 15 τύποι φίλτρων και αυτοί είναι:

Custom Filter:

> Προσαρμοσμένο φίλτρο για καθορισμό 2d filter από τον χρήστη γράφοντας χρησιμοποιώντας GLSL. Στη διαδικασία περιλαμβάνεται οπότε συγγραφή προγράμματος στον Text Editor από τον χρήστη, που θα διευκρινίζει τις δυνατότητες του φίλτρου.

Invert

Sepia

Gray Scale

Prewitt

Sobel

Laplacian

Erosion

Dilation

Sharpen

Blur:

> Όλα τα παραπάνω ονομάζονται ενσωματωμένα φίλτρα. Ρυθμίζονται κατάλληλα ώστε να είναι διαθέσιμα σε μερικά περάσματα, τα οποία έθεσε ο χρήστης στο πεδίο Pass Number.

Motion Blur:

> παράγει εφέ κίνησης στα αντικείμενα. Δίνει τη χαρακτηριστική θαμπάδα στο αντικείμενο που κινείται.

Remove Filter:

> Αφαιρεί το φίλτρο σε κάποιο πέρασμα.

Disable Filter:

Απενεργοποιεί το φίλτρο σε κάποιο πέρασμα .

Enable Filter

Ενεργοποιεί το φίλτρο σε κάποιο πέρασμα.

Ειδικές Επιλογές:

→Pass Number

Πόσες φορές θα χρησιμοποιηθεί το φίλτρο στο αντικείμενο. Το ορίζει ο χρήστης.

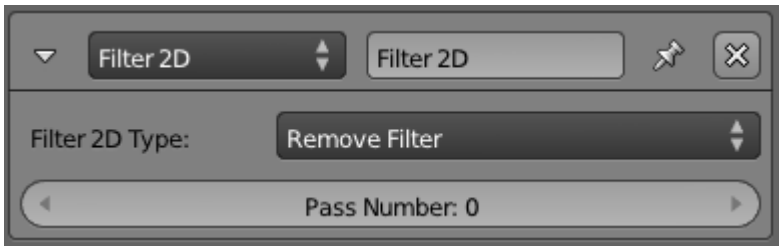

Figure 42 : **2.3.2.5 Filter 2D Actuator**

Game

Αυτός ο ενεργοποιητής επιτρέπει στον χρήστη να εκτελέσει λειτουργίες συγκεκριμένες για το παιχνίδι. Τέτοιες λειτουργίες είναι η Επανεκκίνηση του Παιχνιδιού(Restart Game), ο Τερματισμός του Παιχνιδιού(Quit Game) και η Φόρτωση του Παιχνιδιού(Load Game).

Ειδικές Επιλογές:

→Game

Στο πεδίο αυτό συναντάμε πέντε επιλογές.

Load bge.logic.globalDict

Φόρτωσε bge.logic.globalDict από .bgeconf.

Save bge.logic.globalDict

Σώσε bge.logic.globalDict από .bgeconf.

Quit Game

Μόλις δοθεί θετικός παλμός στον ενεργοποιητή τότε, το blenderplayer βγαίνει από runtime.

Restart Game

Μόλις δοθεί θετικός παλμός στον ενεργοποιητή τότε, το blenderplayer θα επανεκκινήσει το παιχνίδι . Θα φορτώσει ξανά το αρχείο.

Start Game From File

Μόλις δοθεί θετικός παλμός στον ενεργοποιητή τότε, το blenderplayer ξεκινά το .blend file από συγκεκριμένο μονοπάτι.

→File

Στο πεδίο αυτό προσθέτουμε το μονοπάτια από το οποίο θα φορτωθεί το .blend file.

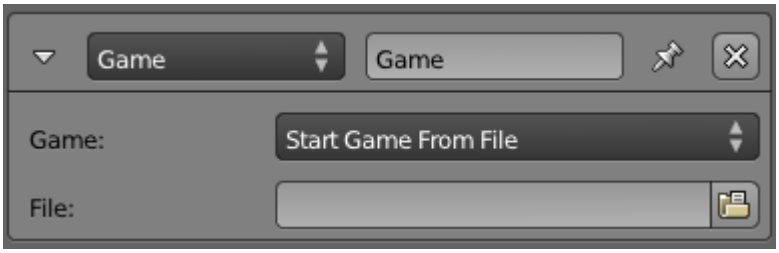

Figure 43 : **2.3.2.6 Game Actuator**

Message

Ο ενεργοποιητής αυτός επιτρέπει στον χρήστη να στέλνει δεδομένα πέρα από τη σκηνή και ανάμεσα στις σκηνές.

Ειδικές Επιλογές:

→To

Στο πεδίο αυτό προστίθεται το αντικείμενο της μετάδοσης. Αν αφεθεί κενό γίνεται μετάδοση παντού (ή αποστολή προς ένα άλλο σκηνικό).

→Subject

Στο πεδίο αυτό προστίθεται το θέμα του μηνύματος. Είναι χρήσιμο αν στέλνονται ορισμένοι τύποι του μηνύματος, όπως ο "end-game", σε έναν αισθητήρα μηνύματος που ακούει για "end game". Δηλαδή για τερματισμό του παιχνιδιού με σύνδεση στον κατάλληλο ενεργοποιητή.

→Body

Το σώμα του κειμένου που στέλνεται (διαβάζεται μόνο από Python).

→Text

Κείμενο στο σώμα διευκρινισμένο από τον χρήστη.

→Property

Ιδιότητα διευκρινισμένη από τον χρήστη.

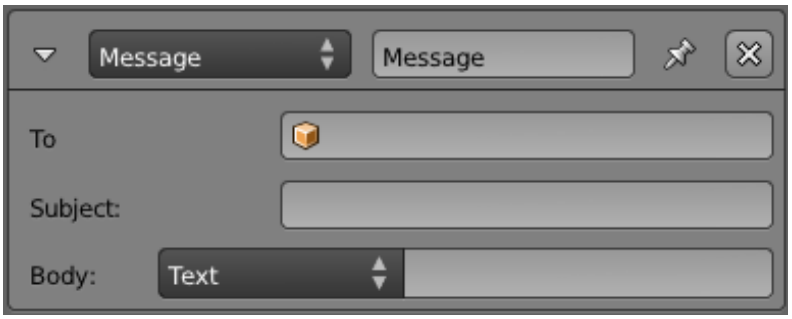

Figure 44 : **2.3.2.7 Message Actuator**

Motion

Με τον ενεργοποιητή Motion θέτουμε ένα αντικείμενο σε κίνηση ή περιστροφή. Μπορούν να συμβούν και τα δύο γεγονότα ταυτόχρονα.
Όσον αφορά τους τρόπους λειτουργίας, συναντάμε δύο. Οι Simple και Servo είναι αυτές οι δύο λειτουργίες που θα κάνουν το αντικείμενο να τηλεμεταφερθεί, περιστραφεί ή κινηθεί δυναμικά.

Ειδικές Επιλογές:

→Motion Type

Ποια από τις δύο λειτουργίες θα επιλεγεί.

Simple Motion

Εφαρμόζει πάνω στο αντικείμενο διαφορετικά είδη κίνησης πάνω στους άξονες.

Servo Control

Μιμείται την κίνηση των αντικειμένων στον φυσικό κόσμο. Σε αυτήν την λειτουργία ο χρήστης θέτει μία ταχύτητα στόχου αλλά και το χρόνο που θα κάνει το αντικείμενο να φτάσει την συγκεκριμένη ταχύτητα. Αφορά την ρύθμιση της δύναμης των αντικειμένων για την επίτευξη της ταχύτητας. Ένα σημείο που πρέπει να τονιστεί είναι πως για να δουλέψει πρέπει στο παράθυρο Physics το αντικείμενο να είναι Dynamic και Actor.

→Simple Motion

Loc

Σε έναν ή περισσότερους από τους τρεις άξονες προσθέτουμε μονάδες Blender. Κάθε φορά που έχουμε θετικό παλμό, ο Motion ενεργοποιείται και

το επιλεγμένο αντικείμενο μετακινείται τόσες μονάδες όσες έχουμε θέσει σε κάθε έναν από τους άξονες X, Y, Z.

Rot

Όταν παραλαμβάνεται θετικός παλμός, το αντικείμενο περιστρέφεται τόσες μονάδες όσες θέσαμε σε κάθε έναν από τους τρεις άξονες.

L

Όταν είναι γκρι, οι συντεταγμένες εφαρμόζονται σφαιρικά. Αντίθετα όταν επιλέξουμε να είναι άσπρο, τότε εφαρμόζονται μόνο τοπικά.

→Servo Control

Reference Object

Στο πεδίο αυτό θα εμφανιστεί μια λίστα με τα αντικείμενα της σκηνής μας. Σύμφωνα με αυτά θα επιλέξουμε ένα το οποίο ο ενεργοποιητής θα χρησιμοποιήσει ως πεδίο αναφοράς για τη μετακίνηση του αντικειμένου. Αν το αφήσουμε κενό τότε θα χρησιμοποιήσει τη παγκόσμια αναφορά.

Linear Velocity

Αφορά την ταχύτητα που θα θέλουμε το αντικείμενο να επιτύχει, σε κάθε έναν από τους τρεις άξονες.

L

Όταν είναι γκρι, οι συντεταγμένες εφαρμόζονται σφαιρικά. Αντίθετα όταν επιλέξουμε να είναι άσπρο, τότε εφαρμόζονται μόνο τοπικά.

X, Y, Z

Θέτει τα μέγιστα και ελάχιστα όρια για τη δύναμη που εφαρμόζεται στο αντικείμενο. Αν τα κουμπιά είναι γκρι, τότε εφαρμόζεται απεριόριστη δύναμη.

Proportional Coefficient

Θέτει τον ανάλογο συντελεστή ο οποίος, θα ελέγξει τις διαφορές μεταξύ της πραγματικής και της γραμμικής ταχύτητας που έχουμε ως στόχο να φτάσει το αντικείμενο.

Integral Coefficient

Θέτει τον ακέραιο συντελεστή ο οποίος, θα ελέγχει την αντίδραση του ποσού των λαθών μέχρις στιγμής στην κίνηση.

Derivative Coefficient

Θέτει τον παράγωγο συντελεστή που ελέγχει τη γενική αντίδραση του αντικειμένου όσον αφορά την κίνηση.

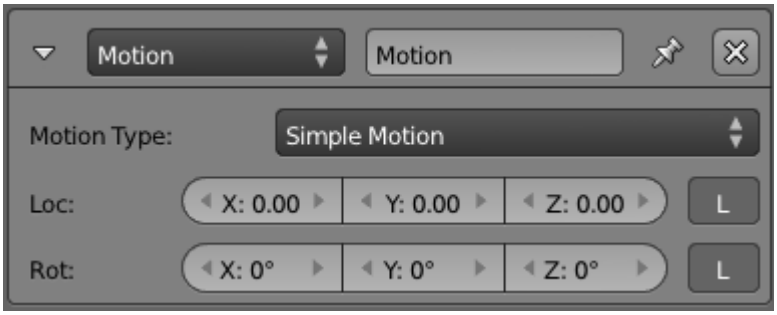

Figure 45 : **2.3.2.8 Motion Actuator**

Parent

Ο συγκεκριμένος ενεργοποιητής αφορά τη σχέση γονέα και επιτρέπει την αλλαγή της, για το επιλεγμένο αντικείμενο.

Ειδικές Επιλογές:

→Scene

Εντοπίζουμε ένα μενού με δύο επιλογές που αφορούν την λειτουργία του γονέα.

Set Parent

Θέτει το συγκεκριμένο αντικείμενο ως γονέα του τρέχοντος αντικειμένου.

Parent Object

Στο πεδίο αυτό θα θέσουμε το όνομα του αντικειμένου γονέα.

Compound

Το σχήμα του αντικειμένου προστίθεται στο σχήμα του γονέα.

Ghost

Όταν το αντικείμενο είναι γονέας και το συγκεκριμένο κουμπί τσεκαρισμένο, τότε το θέτει σε κατάσταση φαντάσματος.

Remove Parent

Αφαιρεί όλους τους γονείς από το συγκεκριμένο αντικείμενο.

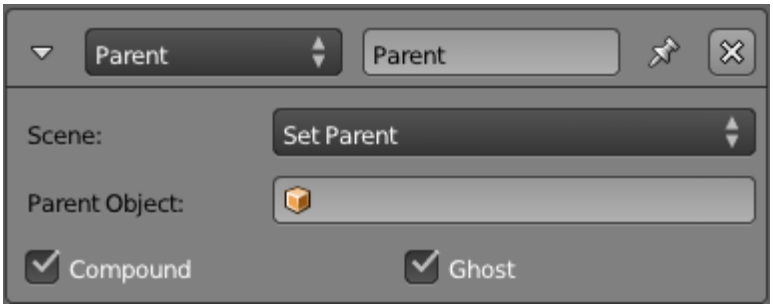

Figure 46 : **2.3.2.9 Parent Actuator**

Property

Ο ενεργοποιητής Property αφορά τις ιδιότητες των αντικειμένων. Μόλις δοθεί θετικό σήμα, έχει το δικαίωμα να αλλάξει τις τιμές των ιδιοτήτων.

Ειδικές Επιλογές:

→Mode

Συναντάμε ένα μενού με τέσσερις επιλογές.

Assign

Μόλις ο ενεργοποιητής δεχτεί ένα θετικό σήμα στην είσοδό του, η ιδιότητα που θέσαμε στο πεδίο Property θα γίνει ίση με την τιμή που θέσαμε στο πεδίο Value.

Add

Μόλις ο ενεργοποιητής δεχτεί ένα θετικό σήμα στην είσοδό του, στην τιμή της ιδιότητα που θέσαμε στο πεδίο Property θα προστεθεί η τιμή που θέσαμε στο πεδίο Value. Αν θέλουμε να τελεστεί αφαίρεση τότε, θέτουμε αρνητική τιμή στο πεδίο Value. Για μεταβλητές Boolean τιμή διαφορετική του μηδενός θα μετράει ως TRUE.

Copy

Από το πεδίο Object διαλέγουμε το αντικείμενο από το οποίο θέλουμε να πάρουμε μια ιδιότητα του(κρυφό πεδίο Property). Μόλις ο ενεργοποιητής ενεργοποιηθεί, θα αντιγράψει αυτή την νέα ιδιότητα στην ιδιότητα του τρέχοντος αντικειμένου.

Toggle

Όταν δεχτεί θετικό σήμα, ο ενεργοποιητής αλλάζει το 0 σε 1, και οποιονδήποτε άλλον αριθμό σε 0. Πολύ χρήσιμο για διακόπτες on/off.

→Property

Στο πεδίο αυτό θέτουμε την ιδιότητα-στόχο στην οποία θα τελεστούν οι αλλαγές.

→Value

Στο πεδίο αυτό θέτουμε την τιμή η οποία θα αλλάξει κατά οποιονδήποτε τρόπο επιλέξουμε την ιδιότητα.

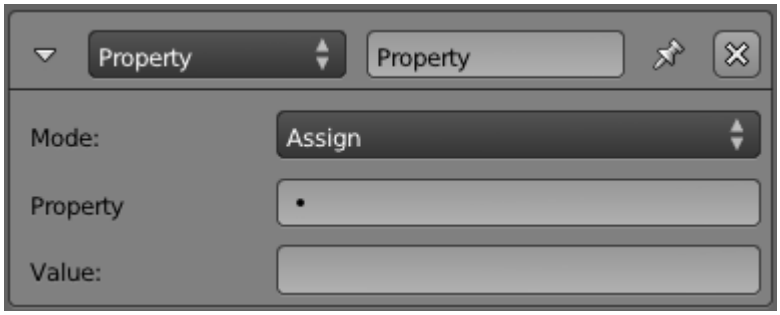

Figure 47 : **2.3.2.10 Property Actuator**

Random

Ο ενεργοποιητής Random βρίσκει τυχαίες τιμές, τις οποίες αναθέτει σε κάποια επιλεγμένη ιδιότητα ενός αντικειμένου.

Ειδικές Επιλογές:

→Seed

Ένας αρχικός αριθμός ο οποίος χρησιμοποιείται για να αρχικοποιήσει τον αλγόριθμο-γεννήτρια τυχαίων αριθμών. Ακτίνα 1-1000.

→Distribution

Ένα μενού με τύπους μεταβλητών, από το οποίο γίνεται η επιλογή τυχαίας τιμής.

Float Neg.Exp.

Καθορίζουμε έναν χρόνο ημιζωής, σύμφωνα με τον οποίο οι τιμές μειώνονται εκθετικά.

Property

Στο πεδίο αυτό προσθέτουμε μια ιδιότητα η οποία πρέπει να είναι Float και από την οποία λαμβάνουμε τιμή.

Half-Life Time

Στο πεδίο αυτό θέτουμε τον χρόνο ημιζωής. Ακτίνα 0.00-10000.00

Float normal

Φυσιολογική διανομή από την οποία επιλέγονται τυχαίοι αριθμοί.

Property

Στο πεδίο αυτό προσθέτουμε μια ιδιότητα η οποία πρέπει να είναι Float και από την οποία λαμβάνουμε τιμή.

Mean

Θέτουμε τον μέσο όρο της κανονικής κατανομής. Ακτίνα -10000.00 έως +10000.00

SD

Θέτουμε την τυπική απόκλιση της κανονικής κατανομής. Ακτίνα 0.00 έως +10000.00

Float uniform

Αφορά τυχαίες τιμές οι οποίες επιλέγονται ομοιόμορφα ανάμεσα σε μια ελάχιστη και μια μέγιστη τιμή, τις οποίες θέτει ο χρήστης.

Property

Στο πεδίο αυτό προσθέτουμε μια ιδιότητα η οποία πρέπει να είναι Float και από την οποία λαμβάνουμε τιμή.

Min

Η ελάχιστη τιμή που θέτει ο χρήστης. Ακτίνα -10000.00 έως +10000.00.

Max

Η μέγιστη τιμή που θέτει ο χρήστης. Ακτίνα -10000.00 έως +10000.00.

Float Constant

Επιστροφή μιας σταθερής τιμής.

Property

Στο πεδίο αυτό προσθέτουμε μια ιδιότητα η οποία πρέπει να είναι Float και από την οποία λαμβάνουμε τιμή.

Value

Η σταθερή τιμή. Ακτίνα 0.00 έως +1.00.

Int Poisson

Εύρεση τυχαίων αριθμών από διανομή Poisson.

Property

Στο πεδίο αυτό προσθέτουμε μια ιδιότητα η οποία πρέπει να είναι Integer και από την οποία λαμβάνουμε τιμή.

Mean

Μέσος όρος κατανομής Poisson. Ακτίνα 0.01 έως +100.00

Int uniform

Αφορά τυχαίες τιμές οι οποίες επιλέγονται ομοιόμορφα ανάμεσα σε μια ελάχιστη και μια μέγιστη τιμή, τις οποίες θέτει ο χρήστης.

Property

Στο πεδίο αυτό προσθέτουμε μια ιδιότητα η οποία πρέπει να είναι Integer και από την οποία λαμβάνουμε τιμή.

Min

Η ελάχιστη τιμή που θέτει ο χρήστης. Ακτίνα -1000.00 έως +1000.00.

Max

Η μέγιστη τιμή που θέτει ο χρήστης. Ακτίνα -10000.00 έως +1000.00.

Int constant

Επιστροφή μιας σταθερής τιμής.

Property

Στο πεδίο αυτό προσθέτουμε μια ιδιότητα η οποία πρέπει να είναι Integer και από την οποία λαμβάνουμε τιμή.

Value

Η σταθερή τιμή. Ακτίνα 0.00 έως +1.00.

Bool Bernoulli

Επιστροφή τυχαίας διανομής, αλλά με καθορισμένη αναλογία από παλμούς TRUE.

Property

Στο πεδίο αυτό προσθέτουμε μια ιδιότητα η οποία πρέπει να είναι Boolean και από την οποία λαμβάνουμε τιμή.

Chance

Θέτουμε το ποσοστό των παλμών TRUE που θα απαιτηθούν.

Bool uniform

Η πιθανότητα να επιλεγεί TRUE ή FALSE είναι 50%.

Property

Στο πεδίο αυτό προσθέτουμε μια ιδιότητα η οποία πρέπει να είναι Integer και από την οποία λαμβάνουμε τιμή.

Bool constant

Επιστροφή μιας σταθερής τιμής.

Property

Στο πεδίο αυτό προσθέτουμε μια ιδιότητα η οποία πρέπει να είναι Integer και από την οποία λαμβάνουμε τιμή.

Value

Θέτουμε την τιμή TRUE ή FALSE.

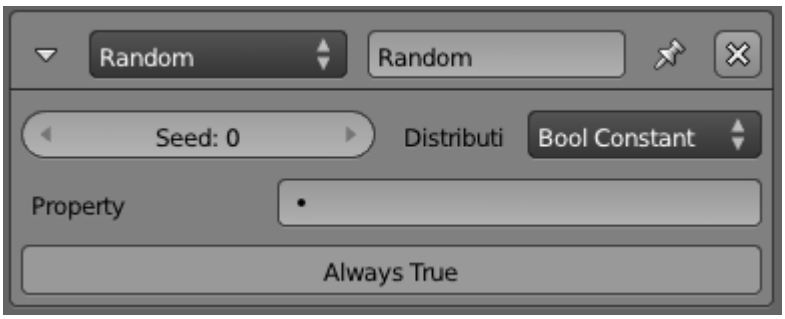

Figure 48 : **2.3.2.11 Random Actuator**

Scene

Αφορά τις σκηνές μέσα στο συγκεκριμένο .blend file. Αυτές μπορούν να χρησιμοποιηθούν ως διαφορετικά επίπεδα ή και για background.

Ειδικές Επιλογές:

→Mode

Συναντάμε ένα μενού με οχτώ επιλογές.

Restart

Ξεκινάει από την αρχή την τρέχουσα σκηνή και όλα ρυθμίζονται ξανά.

Set Scene

Επιλογή μιας σκηνής και αλλαγή με την τρέχουσα.

Set Camera

Αλλάζει την τρέχουσα κάμερα με αυτήν που θα επιλέξουμε.

Add Overlay Scene

Προσθέτει μια νέα σκηνή και την σχεδιάζει μπροστά στην τρέχουσα. Χρήσιμο για πληροφορίες που θέλουμε να είναι πάντα ορατές στην κύρια σκηνή μας.

Add Background Scene

Προσθέτει μια νέα σκηνή και την σχεδιάζει πίσω από την τρέχουσα.

Remove Scene

Αφαιρεί μια σκηνή.

Suspend Scene

Κάνει παύση προσωρινά μόνο μια σκηνή.

Resume Scene

Ξεκινά και πάλι μια σταματημένη σκηνή.

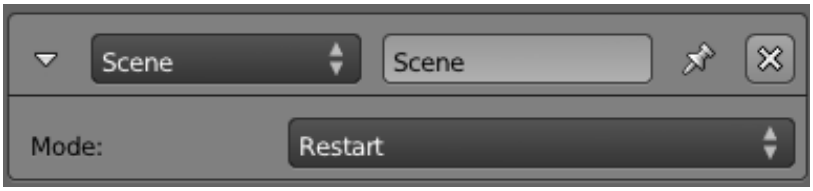

Figure 49 : **2.3.2.12 Scene Actuator**

Sound

Επιλέγει ένα αρχείο ήχου από μία ήδη υπάρχουσα λίστα, ή ανοίγει ένα καινούργιο.

Ειδικές Επιλογές:

→Music File Title

Επιλέγει ένα αρχείο ήχου από την λίστα που εμφανίζεται.

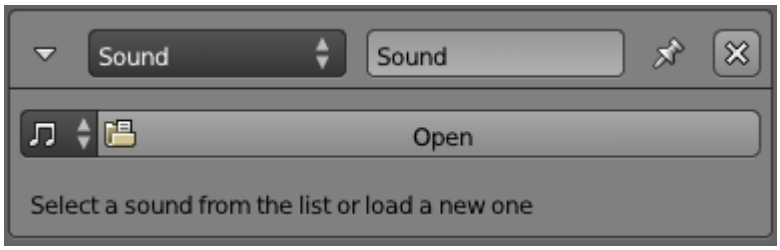

Figure 50 : **2.3.2.13 Sound Actuator**

State

Οι διαδικασίες που τελούνται στη σκηνή μας μπορούν να τοποθετηθούν σε διαφορετικές State. Αυτό βοηθάει το interface να παραμένει καθαρό καθώς τοποθετεί διαφορετικές κατηγορίες διαδικασιών σε διαφορετικές καταστάσεις. Έτσι δημιουργείται μια πιο σύνθετη λογική χωρίς να είναι όμως και μπερδεμένη.

Ειδικές Επιλογές:

→Operation

Υπάρχουν τέσσερις διαφορετικές λειτουργίες.

Change State

Επιλογή μιας κατάστασης και αλλαγή από την τρέχουσα στην καινούργια.

Remove State

Επιλογή μιας κατάστασης και αφαίρεσή της από τις ενεργές καταστάσεις.

Add State

Επιλογή μιας κατάστασης και προσθήκη της στις ενεργές καταστάσεις.

Set State

Επιλογή μιας νέας κατάστασης και μετακίνηση σε αυτήν, απενεργοποιώντας όλες τις πρόσθετες καταστάσεις.

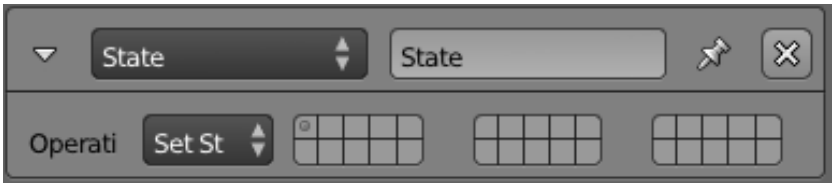

Figure 51 : **2.3.2.14 State Actuator**

Visibility

Σε κατάσταση runtime ο ενεργοποιητής αυτός επιτρέπει τον χρήστη να αλλάξει την ορατότητα των αντικειμένων.

Ειδικές Επιλογές:

→Visible

Καθιστά το αντικείμενο ορατό

→Occlusion

Για τη συγκεκριμένη επιλογή πρέπει να έχει προηγηθεί αρχικοποίηση από την καρτέλα Physics.

→Children

Θέτει visible και occlusion σε όλα τα αντικείμενα-παιδιά , αλλά και αναδρομικά στα παιδιά των παιδιών.

Figure 52 : **2.3.2.15 Visibility Actuator**

ΚΕΦΑΛΑΙΟ 3

TURRET DEFENCE GAME

Για να καταλάβουμε καλύτερα τον τρόπο που δουλεύει το Game Engine του BLENDER, θα δημιουργήσουμε ένα παιχνίδι που θα βασίζεται στα Logic Bricks, σε Properties, με δόσεις animation, κώδικα Python αλλά και διάφορες άλλες επιλογές.

Το παιχνίδι θα ονομάζεται Turret Defense και η κεντρική ιδέα είναι η απώθηση και εξόντωση των εχθρών. Στη μέση της σκηνής θα βρούμε τοποθετημένο ένα κανόνι το οποίο θα επιτίθενται από τριγύρω διάφοροι εχθροί(δημιουργημένα αρχικά ως απλά cubes). Σκοπός είναι να τα πυροβολήσουμε πριν προλάβουν φτάσουν το κανόνι, αφού έχει μόνο τρεις ζωές.

Το παιχνίδι αν και σχετικά απλό, δίνει τη δυνατότητα στον δημιουργό του να κατανοήσει πως λειτουργεί ο Logic Editor του Blender, να χρησιμοποιήσει τα Logic Bricks, τα Properties και στο τέλος να είναι ικανός να δημιουργήσει ένα νέο δικό του παιχνίδι από την αρχή.

Παράλληλα θα υπάρξει ενασχόληση και με την προσθήκη textures, την κατασκευή σχημάτων, τον προγραμματισμό, την ρύθμιση κάμερας αλλά και την δημιουργία Splash Screen ώστε να εμπλουτιστεί η εκμάθηση του Game Engine του Blender.

3.1 ΚΑΤΑΣΚΕΥΑΖΟΝΤΑΣ ΕΝΑ ΑΠΛΟ ΚΑΝΟΝΙ

Αρχικά πρέπει να αλλάξουμε το mode από Blender Render σε Blender Game όπως δείξαμε προηγουμένως. Με αυτόν τον τρόπο γίνονται διαθέσιμες όλες οι δυνατότητες της παιχνιδομηχανής.

Ξεκινάμε λοιπόν, με έναν κύβο στη μέση της σκηνής μας, ο οποίος θα τροποποιηθεί καταλλήλως και θα γίνει ο πυργίσκος μας. Για να έχουμε μια καλύτερη άποψη του κύβου μας την ώρα που τον μετατρέπουμε, πατάμε το νούμερο 7 από το numpad. Η κίνηση αυτή μας δίνει την δυνατότητα να τον βλέπουμε από την επάνω όψη του. Για να του δώσουμε το σχήμα ενός κανονιού θα πρέπει να τον επεκτείνουμε.

- Με το πάτημα του tab επιλέγουμε τις άκρες του κύβου. Όπως βλέπουμε είναι όλες επιλεγμένες(πορτοκαλί πλαίσιο). Ο στόχος μας όμως είναι να προσδιορίσουμε αλλά και να επιλέξουμε αυτές που εμείς θέλουμε, ώστε στη συνέχεια να τις τροποποιήσουμε. Με το κουμπί A το αποεπιλέγουμε.

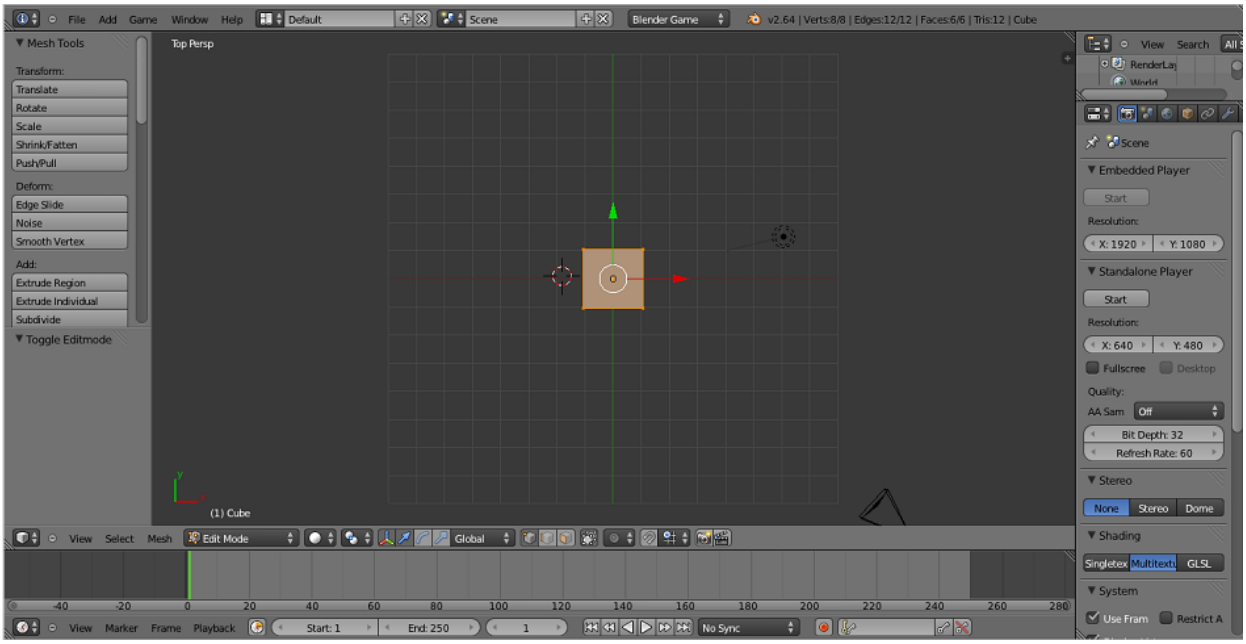

Figure 53 : **3.1.1 Επιλογή του κύβου με TAB**

- Με το πάτημα του B και με κλικ του mouse επιλέγουμε τις κατάλληλες άκρες.
- Με το πάτημα του E κάνουμε extrude. Δηλαδή εξωθούμε τις άκρες σύμφωνα με τον άξονα y.

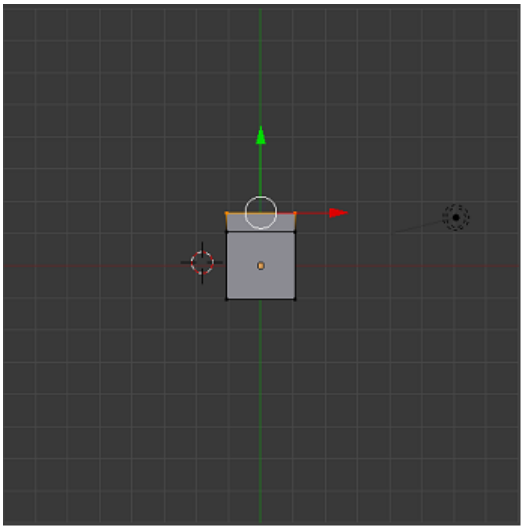

Figure 54 : **3.1.2 Extrude τον κύβο**

- Με το S κάνουμε scale τις άκρες που εξωθήσαμε προς τα μέσα, ώστε να φτιάξουμε την βάση μιας κάννης όπλου.

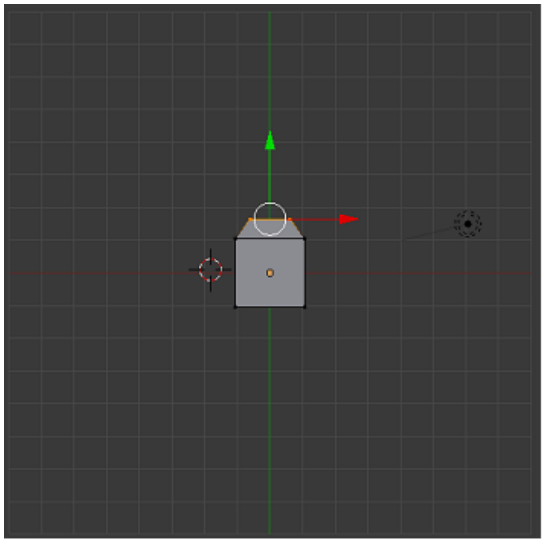

Figure 55 : **3.1.3 Scale τον κύβο**

- Με το E ξανατραβάμε τις άκρες, αρκετά μακριά, ώστε να δώσουμε την αίσθηση μιας κανονικής κάννης, σύμφωνα πάντα με τον άξονα y.
- Δεξιά από το ανοιχτό μενού Properties διαλέγω την καρτέλα Object και μετονομάζω το αντικείμενο σε Turret προς διευκόλυνσή μας.

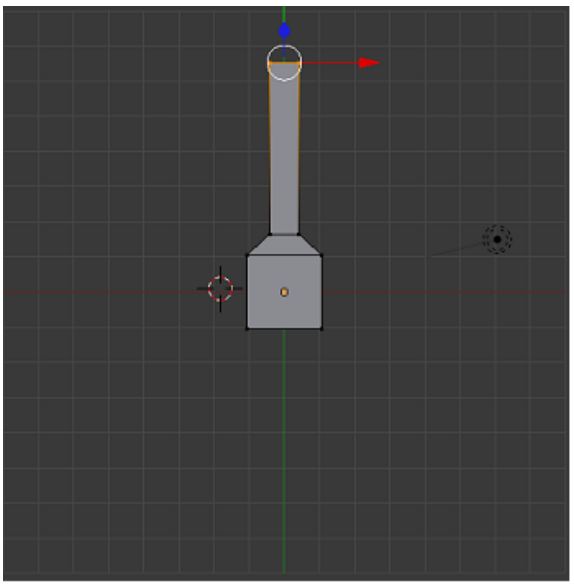

Figure 56 : **3.1.4 Extrude για τη δημιουργία κάννης**

Το κανόνι-πυργίσκος που μόλις δημιουργήσαμε θέλουμε να αποκτήσει κάποιες δυνατότητες. Αρχικά θα υλοποιήσουμε την περιστροφή γύρω από τον εαυτό του. Με αυτόν τον τρόπο θα μπορεί να εξουδετερώνει τα εχθρικά κιβώτια από όποια μεριά και αν έρχονται.

Αυτό που χρειάζεται είναι να αλλάξουμε το μενού σε Logic Editor.

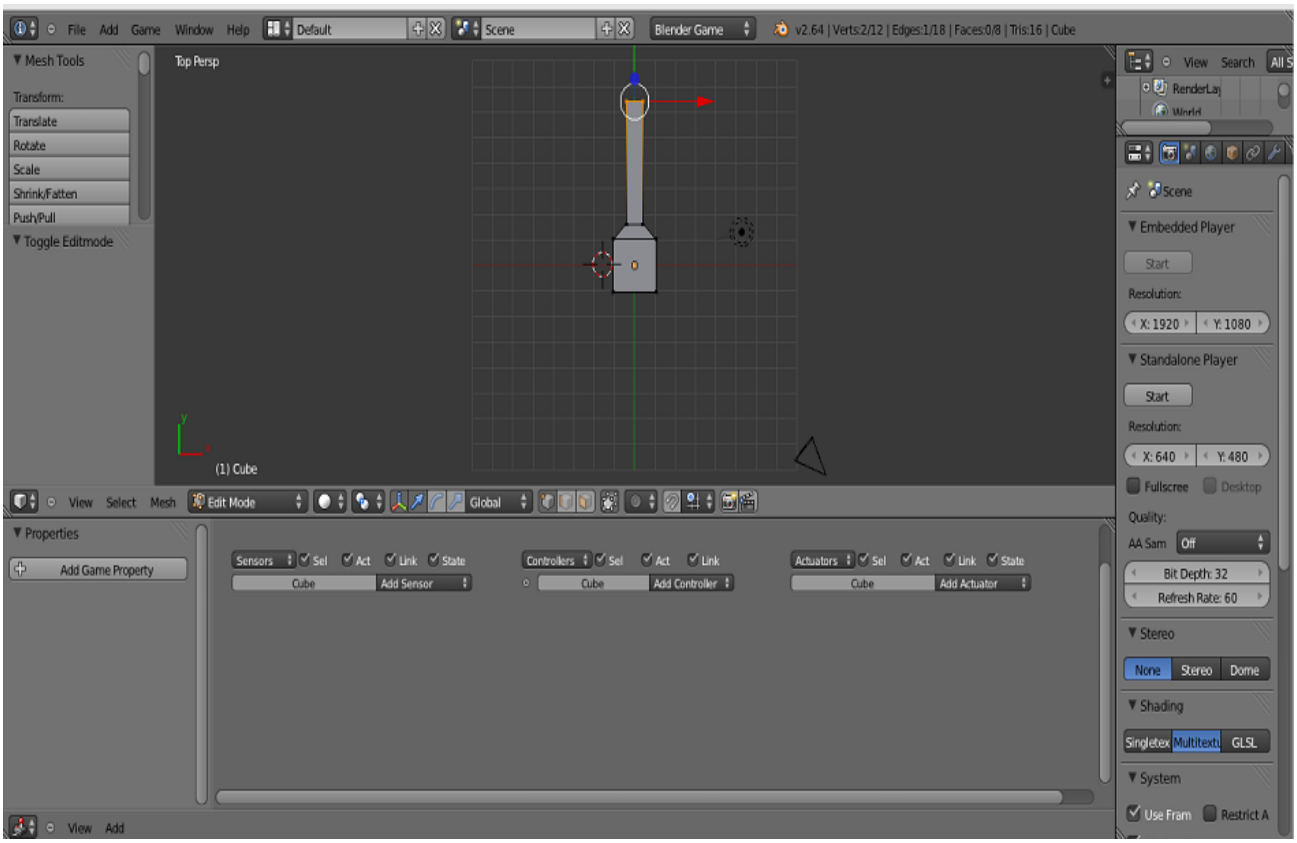

Figure 57 : 3.1.5 Logic Editor

Περιστροφή του κανονιού προς τα αριστερά:

- Add Sensor: Keyboard – Στο πεδίο key κάνω click και πατάω το αριστερό βελάκι ώστε να δω την τιμή Left Arrow.
- Add Controller: Or – Επιτυγχάνει τη σύνδεση μεταξύ Sensor και Actuator.
- Add Actuator: Motion – Για να δώσουμε τον παλμό που χρειάζεται προκειμένου να αρχίσει να γυρνάει πρέπει να ασχοληθούμε με το πεδίο Rot. Ο άξονας z είναι αυτός που μας ενδιαφέρει στην προκειμένη περίπτωση, οπότε θα θέσουμε την τιμή 0,5 .
- Ενώνουμε τα Logic Bricks μεταξύ τους ώστε να αλληλεπιδρούν.

Περιστροφή του κανονιού προς τα δεξιά:

- Add Sensor: Keyboard – Στο πεδίο key κάνω click και πατάω το δεξί βελάκι ώστε να δω την τιμή Right Arrow.
- Add Controller: Or – Επιτυγχάνει τη σύνδεση μεταξύ Sensor και Actuator.
- Add Actuator: Motion – Στο πεδίο Rot στον άξονας z θέτουμε την τιμή -0,5 ,ώστε να γυρίσει από τη δεξιά μεριά.
- Ενώνουμε τα Logic Bricks μεταξύ τους ώστε να αλληλεπιδρούν.

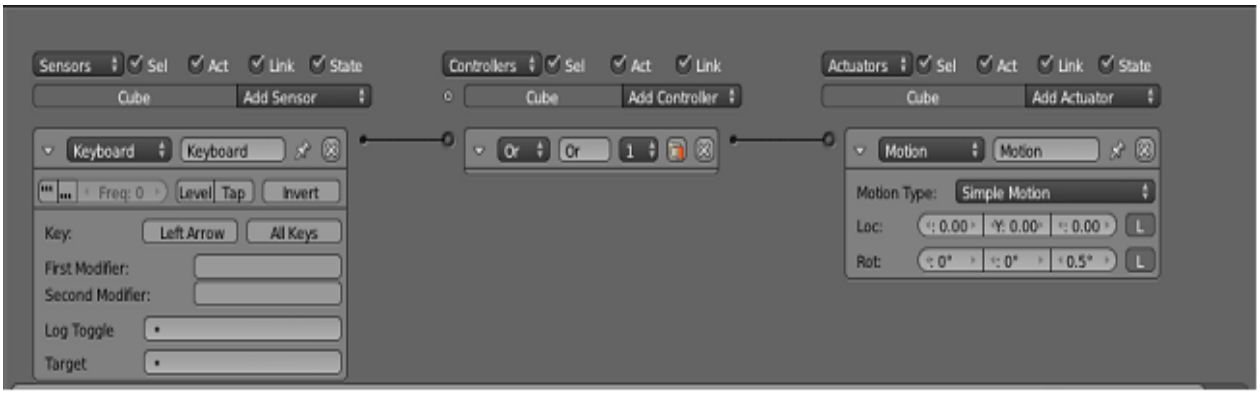

Figure 58 :3.1.6 Keyboard Sensor για δημιουργία κίνησης με το αριστερό βελάκι(αντίστοιχα και στο δεξί)

3.2 ΔΗΜΙΟΥΡΓΙΑ ΚΑΙ ΕΚΤΟΞΕΥΣΗ ΣΦΑΙΡΑΣ

Η περιστροφή έχει ολοκληρωθεί και το επόμενο βήμα είναι η δημιουργία της σφαίρας την οποία θα πετάει το κανόνι μας προς τα εχθρικά κιβώτια. Στη μέση του κανονιού θα τοποθετήσουμε λοιπόν μια Icosphere. Για την δημιουργία πατάμε Shift+A και στο μενού που μας εμφανίζεται διαλέγουμε Mesh→Icosphere. Από την καρτέλα Object του μενού Properties την μετονομάζουμε σε Bullet μιας και που θα αναλάβει χρέη σφαίρας όπλου-κανονιού.

Παρατηρούμε πως η σφαίρα μας φαίνεται αρκετά μεγάλη για να χωρέσει στην κάννη του Turret. Δεν υπάρχει όμως πρόβλημα, αφού θέλουμε απλά να δημιουργήσουμε την ψευδαίσθηση μιας σφαίρας που εκτοξεύεται. Για αυτό το λόγο δεν κάνουμε καθόλου αλλαγές στο σχήμα ή το μέγεθός της. Η λογική που θα ακολουθήσουμε είναι να αναπαράγουμε κάποια αντίγραφά της στην άκρη της κάννης και να τα εκτοξεύουμε. Για να επιτευχθεί αυτό θα χρειαστεί να δουλέψουμε με κάποια άλλα αντικείμενα. Έτσι λοιπόν η σφαίρα μας πρέπει να μεταφερθεί σε ένα άλλο layer Επιλέγουμε λοιπόν με δεξί κλικ την σφαίρα και από την καρτέλα Object → Move to Layer → επιλογή του δεύτερου layer από το σχήμα που εμφανίζεται.

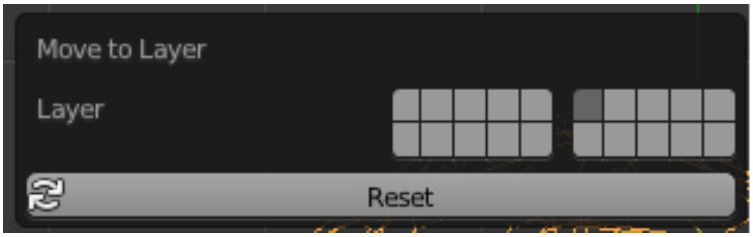

Figure 59 : **3.2.1 Παράθυρο μετακίνησης αντικειμένου σε άλλο layer**

Συνεχίζοντας θα δώσουμε στη σφαίρα μας κάποιες φυσικές ιδιότητες. Από την καρτέλα Physics του μενού Properties στα δεξιά κάνουμε τις εξής κινήσεις:

- Physics→ Physics Type→ Dynamic
  Μετατρέποντας το αντικείμενο από static σε dynamic του δίνουμε φυσικές ιδιότητες. Η βαρύτητα θα μπορεί να το έλκει, θα αποκτήσει μάζα, θα αντιδράει στην τριβή και θα αλληλεπιδρά με τα υπόλοιπα αντικείμενα.
- Collision Bounds(check)→ Bounds→ Sphere
  Η συγκεκριμένη ιδιότητα δίνεται από το Blender για το χειρισμό των σχημάτων των αντικειμένων. Τα όρια-φράγματα του αντικειμένου υπολογίζονται από το κέντρο του. Ορίζουμε ένα σχήμα κοντά σε αυτό που έχουμε ώστε η αλληλεπίδρασή του με το περιβάλλον να είναι ανάλογη. Π.χ. στην συγκεκριμένη περίπτωση θέλουμε να φέρεται ως σφαίρα και το περιβάλλον να το αναγνωρίζει ως μια τέτοια.

Figure 60 : **3.2.2 Καρτέλα Physics**

Στη συνέχεια επανερχόμαστε στο πρώτο layer για να συνδέσουμε τη σφαίρα με το Turret και να προκαλέσουμε αναπαραγωγή σφαιρών. Για να την δημιουργία των σφαιρών θα χρησιμοποιήσουμε το αντικείμενο Empty(Shift+A για εμφάνιση του μενού και έπειτα επιλογή του αντικειμένου).

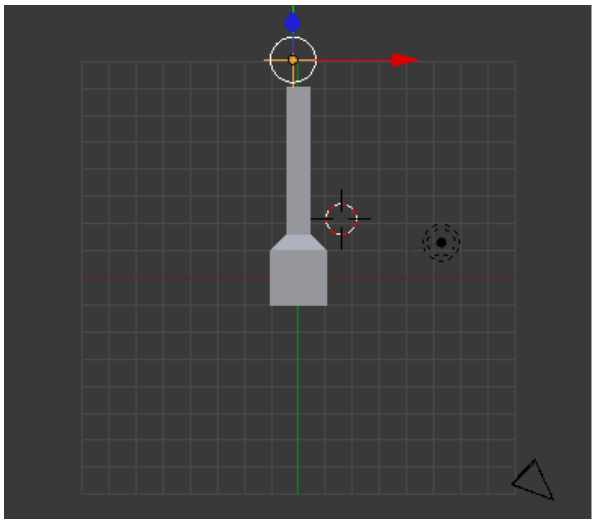

Figure 61 : **3.2.3 Προσθήκη αντικειμένου Empty στην άκρη της κάννης**

Το Empty είναι ένα κενό αντικείμενο, χωρίς Γεωμετρία και χρησιμοποιείτε για διάφορους βοηθητικούς λόγους. Έχει πολλές δυνατότητες που το καθιστούν εύχρηστο και χρήσιμο. Μπορεί να γίνει parent με οποιοδήποτε από τα υπόλοιπα αντικείμενα και με αυτόν τον τρόπο να τα ελέγχει. Επίσης είναι ικανό να κρατάει μια θέση και να 'καλεί' άλλα αντικείμενα, όπως και στην περίπτωσή μας.

Ο στόχος μας όμως είναι να δώσουμε την ψευδαίσθηση πως οι σφαίρες εκτοξεύονται από την άκρη της κάννης μας. Διαλέγουμε το Empty με δεξί κλικ και με το κουμπί G το μετακινούμε στο σημείο που θέλουμε. Το μετονομάζουμε σε BulletSpawn.

Για τη σύνδεση του Empty με την Bullet του δεύτερου layer θα χρειαστεί να επανέλθουμε στο Logic Editor.

- Add Sensor: Keyboard – Στο πεδίο key κάνω click και πατάω το spacebar ώστε κάθε φορά που το πατάμε στο παιχνίδι να εκτοξεύουμε μια σφαίρα.
- Add Controller: Or – Επιτυγχάνει τη σύνδεση μεταξύ Sensor και Actuator.
- Add Actuator: Edit Object→Add Object→Object→Bullet – Στο πεδίο Linear Velocity στον άξονα y ορίζουμε τη ταχύτητα με την οποία θα εκτοξεύεται η σφαίρα. Θέτουμε την τιμή 20. Αυτό που παρατηρούμε είναι το κουμπί L απενεργοποιημένο. Οι αλλαγές λοιπόν αφορούν τον άξονα y σε παγκόσμιο επίπεδο. Το γεγονός αυτό όμως θα αποτρέψει τις σφαίρες μας να ακολουθούν την κατεύθυνση της περιστροφής και να μένουν κολλημένες στην κατεύθυνση του παγκόσμιου άξονα y. Επιλέγουμε λοιπόν το L και τώρα κάθε φορά που ο άξονας y αλλάζει τοπικά, θα αλλάζει και η κατεύθυνση της σφαίρας.
- Ενώνουμε τα Logic Bricks μεταξύ τους ώστε να αλληλεπιδρούν.

Με μια πρώτη δοκιμή του παιχνιδιού με το κουμπί P, παρατηρούμε πως η σφαίρα πράγματι εκτοξεύεται αλλά δεν συνοδεύει το Turret στην περιστροφή του. Για να το διορθώσουμε πρέπει να ορίσουμε το Empty ως 'child' του Turret.

- Δεξί κλικ στο Empty για να το επιλέξουμε και έπειτα Shift+δεξί κλικ στο Turret. Στη συνέχεια Control+P και επιλογή του Object από το παράθυρο που εμφανίζεται.

Από δω και πέρα όταν θα κινείται το κανόνι, ταυτόχρονα θα κινείται και το Empty.

3.3 ΡΥΘΜΙΣΗ CAMERA, 3D VIEW

Σε αυτό το σημείο θα προσθέσουμε ένα plane στο χώρο μας, ώστε να μπορούμε να δουλεύουμε σε ένα επίπεδο που θα οριοθετήσει και τον χώρο μας. Για την προσθήκη πατάμε Shift+A→ Mesh→Plane . Με το S μπορούμε να το μεγαλώσουμε ώστε να πιάνει έναν ικανοποιητικό χώρο στο περιβάλλον μας. Μετονομάζουμε το plane από την καρτέλα Object σε Ground. Ίσως χρειαστεί να μετακινήσουμε το plane σε θέση που να μην καλύπτει το Turret μας, αλλά αυτό εξαρτάται από τν αρχική του τοποθέτηση.

Αν και το παιχνίδι μας λειτουργεί δεν μπορούμε ακόμη να καταλάβουμε τις 3d δυνατότητες του Blender, διότι βλέπουμε το αντικείμενό μας μόνο από την πάνω μεριά. Οι κάμερες θα μας βοηθήσουν στην 3d view του παιχνιδιού με την κατάλληλη ρύθμισή τους.

Αρχικά θα ήταν καλό να δημιουργήσουμε μια First Perspective View ώστε ο παίκτης να αισθάνεται πως βρίσκεται ο ίδιος πίσω από το κανόνι. Πρώτο βήμα επομένως είναι ο χωρισμός του 3d παραθύρου στα δύο, κάνοντας κλικ και σέρνοντας το ποντίκι στις διακεκομμένες άσπρες γραμμές της επάνω δεξιά γωνίας. Το αριστερά παράθυρο θα χρησιμοποιηθεί ως οδηγός για το ορατό πεδίο της κάμερας, οπότε με το πλήκτρο 0 μεταφερόμαστε σε Camera Perspective View. Δουλεύουμε επομένως στο δεξί παράθυρο. Με το πλήκτρο 1 μεταφερόμαστε σε Front Perspective View και επιλέγουμε την κάμερά μας με δεξί κλικ. Με το κουμπί G θα μετακινήσουμε την κάμερά μας και με το R θα την περιστρέψουμε στον άξονα z κατάλληλα, έως ότου στο αριστερό παράθυρο να έχουμε την παρακάτω εικόνα.

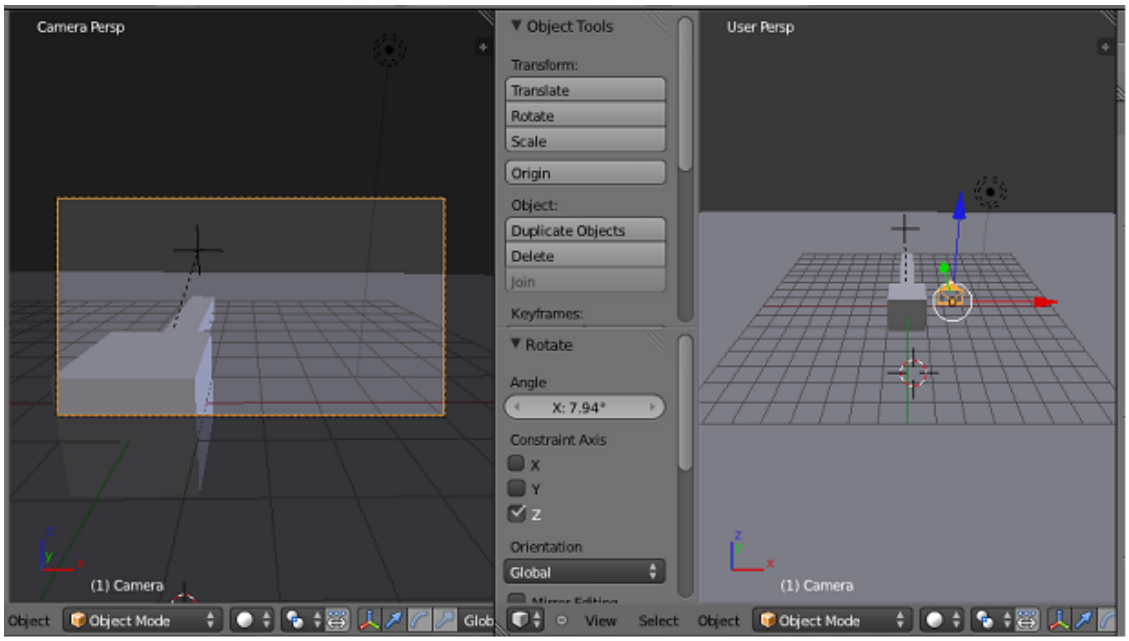

Figure 62 : 3.3.1 Διπλό παράθυρο, κοντινή λήψη

Ξαναγυρνάμε σε μονό παράθυρο σέρνοντας τις διακεκομμένες γραμμές προς την αντίθετη από πριν κατεύθυνση. Με δεξί κλικ επιλέγουμε πρώτα την κάμερα, και με Shift+δεξί κλικ το Turret. Με Control+P θέτουμε ως Parent το Turret στην κάμερα, ώστε η δεύτερη να το ακολουθεί καθώς περιστρέφεται.

Όταν το παιχνίδι θα γίνει εκτελέσιμο αρχείο θα θέλουμε ο χρήστης να έχει τη δυνατότητα να βλέπει τη σκηνή από δύο πλευρές. Μία ως First Person Shooter και μία από ψηλά για να ελέγχουμε το περιβάλλον μας. Αυτές οι δύο κάμερες θα πρέπει να εναλλάσσονται με τη πληκτρολόγηση των αριθμών 1 και 2.

Αρχικά μετονομάζουμε την κάμερά μας σε CameraFPS. Στον Logic Editor προσθέτουμε:

- Add Sensor→ Keyboard – κλειδί το κουμπί 1.
- Add Controller→ Or
- Add Actuator→ Scene – Mode→ Set Camera – Camera Object→ CameraFPS

Με αυτόν τον τρόπο θέσαμε την εναλλαγή σε κάμερα FPS όταν πατιέται ο αριθμός 1.

Στη συνέχεια με Shift+D αντιγράφουμε την κάμερα και προεκτείνουμε τη νέα ψηλά και έξω από τα όρια της σκηνής μας. Ο συνδυασμός των πλήκτρων που ακολουθήσαμε δεν αντιγράφει μόνο την κάμερα αλλά και τα λογικά τούβλα και όποιες ιδιότητες θα μπορούσαμε να είχαμε θέσει. Τη μετονομάζουμε σε CameraTPS (Third Person Shooter). Όσον αφορά τα Logic Bricks το μόνο που πρέπει να αλλάξουμε είναι ο αριθμός 1 στον Keyboard Sensor ο οποίος θα γίνει 2.

Για να μπορούμε να δούμε τη σκηνή μας από την νέα μας κάμερα πρώτα την επιλέγουμε και έπειτα πατάμε Control+0. Όπως ακριβώς και με τη πρώτη μας κάμερα, έτσι και εδώ θα χωρίσουμε το παράθυρο στα δύο και από το δεξί μέρος θα κουνάμε τη κάμερα μέχρι το αριστερά να έχουμε τη θέα που θέλουμε. Όσο βρισκόμαστε ακόμη στην CameraTPS διαλέγουμε την καρτέλα Object Data από το μενού Properties δεξιά από τη σκηνή μας και στο πεδίο Clipping θέτουμε τελική τιμή 700.000 ώστε να μπορούμε να έχουμε περισσότερο βάθος στη σκηνή μας.

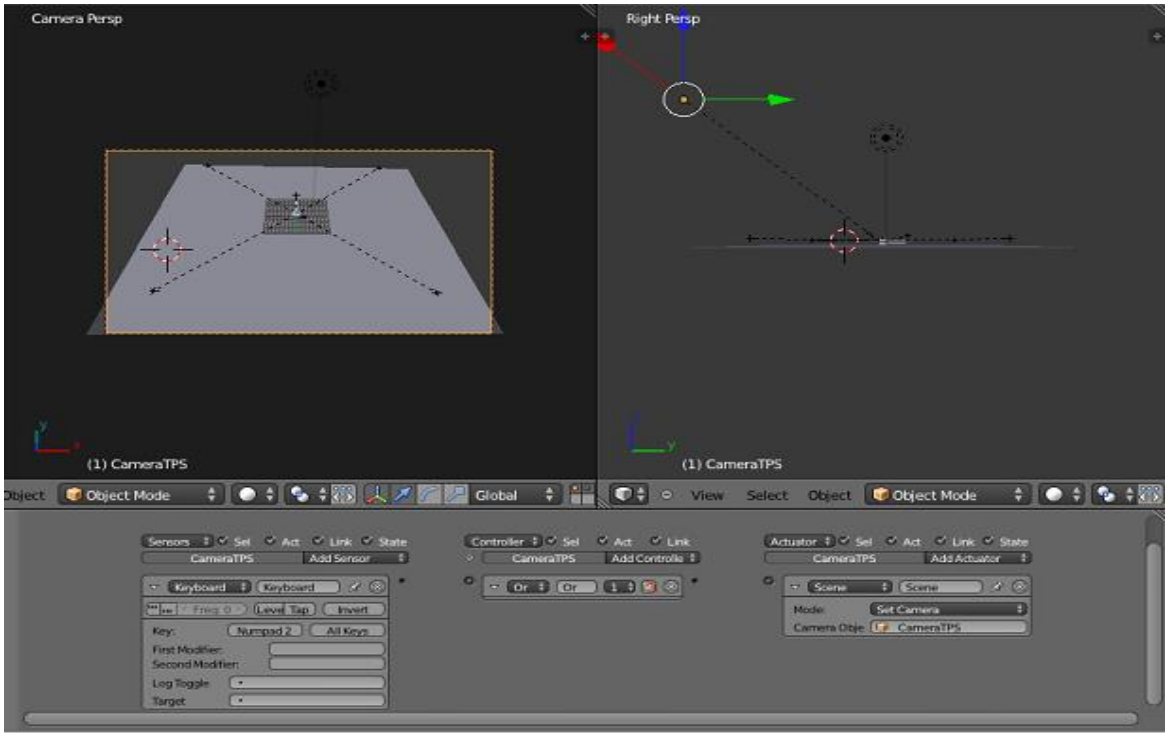

Figure 63 : **3.3.3 Διπλό παράθυρο, μακρινή λήψη**

3.4 ANIMATION ΟΠΙΣΘΟΔΡΟΜΗΣΗΣ TURRET ΚΑΤΑ ΤΗΝ ΕΚΤΟΞΕΥΣΗ ΣΦΑΙΡΑΣ

Όταν εκτοξεύουμε μια σφαίρα θέλουμε ταυτόχρονα να συμβαίνει κίνηση οπισθοδρόμησης στον πυργίσκο, κάτι το οποίο πετυχαίνεται με τη δημιουργία του ανάλογου animation.

Πρώτο βήμα η μεταφορά από το Default screen layout στο οποίο βρισκόμαστε τώρα, σε Animation layout, από γραμμή εργαλείων που βρίσκουμε στο επάνω μέρος του παραθύρου. Το παράθυρό μας έχει πλέον χωριστεί στα δυο. Από αριστερά βλέπουμε τον χώρο στον οποίο θα δουλέψουμε το animation. Θα ασχοληθούμε φυσικά με keyframes. Τα keyframes είναι τα χαρακτηριστικά καρέ. Σε κάθε ένα από αυτά συμβαίνει μία μεμονωμένη κίνηση του αντικειμένου μας, ενώ όταν αναπαράγονται όλα μαζί στη σειρά, έχουμε ένα animation.

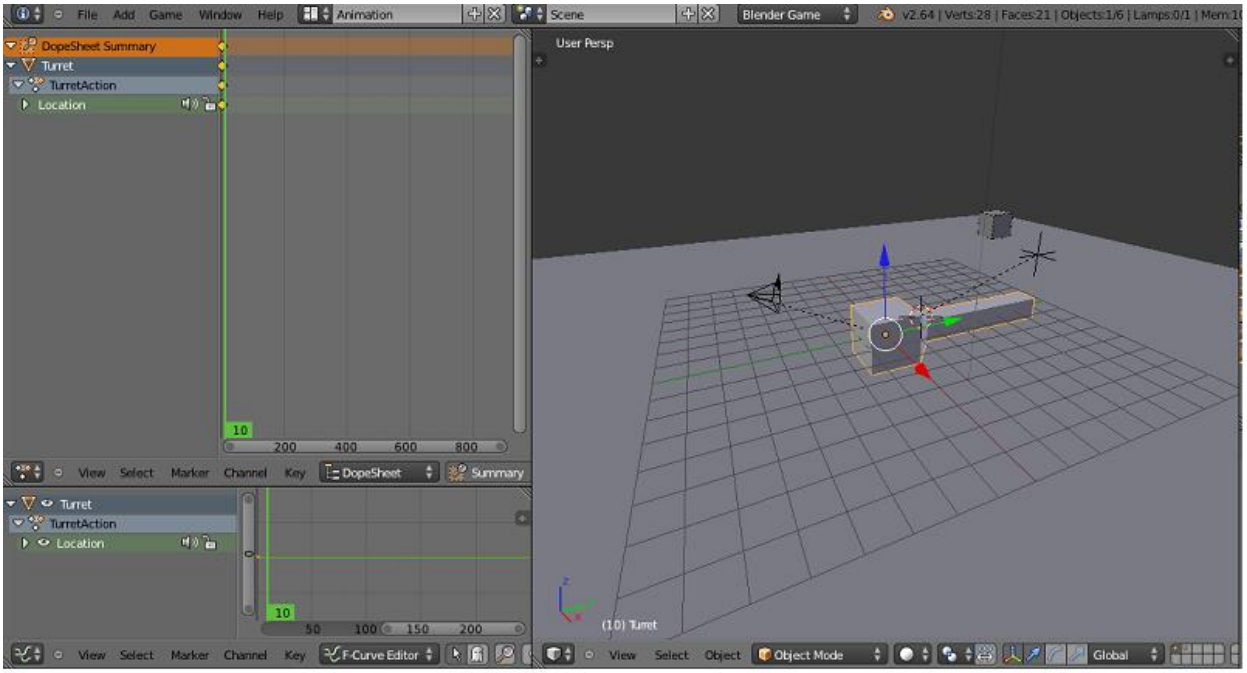

Figure 64 : 3.4.1 Παράθυρο animation

Θα παρατηρήσουμε ότι από προεπιλογή είναι επιλεγμένος ο DopeSheet Editor. Ο συγκεκριμένος Editor, που έχει τις απαρχές του στους animators που σχεδίαζαν στο χέρι, δίνει την δυνατότητα στον σχεδιαστή να έχει μια συνολική εικόνα της σκηνής του, παρατηρώντας οποιαδήποτε αλλαγή συμβαίνει. Αυτή η αλλαγή μπορεί να αφορά την κάμερα, τα αντικείμενα, τον ήχο κτλ.

Η διαδικασία δημιουργίας του animation έχει ως εξής:

- Δεξί κλικ και επιλογή του πυργίσκου.
- Με το κουμπί I ανοίγει η καρτέλα για την εισαγωγή του keyframe που αντιστοιχεί σε αυτό που θέλουμε να δημιουργήσουμε. Στην περίπτωσή μας θέλουμε στιγμιαία αλλαγή της τοποθεσίας του πυργίσκου άρα επιλέγουμε το Location.
- Για να είναι σωστή η κίνηση θα δώσουμε τον αριθμό των 10 frames στην θέση που φαίνεται παρακάτω. Με αυτόν τον τρόπο η κίνηση προς τα πίσω θα κρατήσει για 10 frames.
- Με ήδη επιλεγμένο το Turret μας το μετακινούμε με το G στον άξονα y προς τα πίσω και έπειτα με το πάτημα του I κλειδώνουμε τη νέα location.
- Για να δούμε μια προεπισκόπηση αρκεί να πατήσουμε Alt+A. Escape για επαναφορά.
- Σε αυτό το σημείο θέλουμε να πετύχουμε την επαναφορά του Turret στην αρχική του θέση με πιο αργό ρυθμό από αυτόν με τον οποίο μετακινήθηκε προς τα πίσω.

- Η αρχική θέση μας βρίσκεται στα πρώτα keyframe. Άρα το μόνο που έχουμε να κάνουμε είναι με το B να τα επιλέξουμε και με το Shift+D να τα αντιγράψουμε στο frame 30. Δηλαδή διπλασιάσαμε το χρόνο με τον οποίο θα γίνεται η απόσβεση της κίνησης.

Μόλις τελειώσουμε επανερχόμαστε στο Default Scene View. Για να ενσωματώσουμε το animation στη σκηνή μας θα δουλέψουμε με τα logic bricks. Στο αντικείμενο Turret θα προσθέσουμε τον Sensor→ Keyboard, ο οποίος θα ανταποκρίνεται στο Spacebar. Το Motion που θα δουλέψουμε είναι ο Action (για παλαιότερες εκδόσεις είναι ο F-Curve). Για Start Frame θα εισάγουμε το 1, ενώ για End Frame το 30. Στο πεδίο Value θα επιλέξουμε το TurretAction. Επίσης ενεργοποιούμε το ADD καθώς και το L, ώστε οι αλλαγές να συμβαίνουν στους τοπικούς άξονες.

Ένα πρόβλημα που αντιμετωπίζουμε βρίσκεται στο First Person Perspective κατά το οποίο δεν είναι εμφανής η οπισθοδρόμηση. Για τη λύση αυτού του προβλήματος προσθέτουμε ένα Empty στην μέση του Turret και το ονομάζουμε TurretCore. Ο πυρήνας λοιπόν θα κρατάει από εδώ και πέρα τα bricks που αφορούν τις κινήσεις του πυργίσκου. Για να γίνει αυτό αρκεί να τις αντιγράψουμε.

- Επιλογή του TurretCore, Shift+Επιλογή του Turret
- Από το 3d View Menu ανοίγουμε το Object Menu και επιλογή Game→ Copy Logic Bricks
- Από το TurretCore διαγράφουμε τα Bricks που αφορούν το animation ενώ από το Turret τα Bricks που αφορούν την κίνηση.
- Κάνουμε parent το Turret στο TurretCore
- Κάνουμε parent την κάμερα στο TurretCore

Με το πάτημα του Spacebar έχουμε μια οπισθοδρόμηση και εκτόξευση σφαίρας. Η σωστή χρήση της οπισθοδρόμησης έγκειται στο γεγονός πως πραγματοποιείται μία φορά κατά την εκτόξευση μίας και μόνο σφαίρας. Αυτό που θα προσέξουμε είναι πως αν συνεχίσουμε να πατάμε το Spacebar θα έχουμε πολλαπλές εκτοξεύσεις σφαιρών κατά τη διάρκεια της οπισθοδρόμησης. Πρέπει επομένως να περιορίσουμε τις εκτοξεύσεις σε μία κατά τη διάρκεια μίας οπισθοδρόμησης.

Αρχικά θα θέσουμε το Turret υπεύθυνο για την ενεργοποίηση αναπαραγωγής σφαιρών στο BulletSpawn. Στα Logic Bricks θα προσθέσουμε τον Actuator-Message ο οποίος θα στέλνει στο BulletSpawn ένα μήνυμα Spawn. Θα τον ενώσουμε με τον Controller που ανταποκρίνεται στον Sensor Keyboard και ορίζει το πάτημα του Spacebar. Ο BulletSpawn πρέπει να δέχεται αυτό το μήνυμα. Στα Logic Brick του επομένως θα διαγράψουμε τον Keyboard Sensor και θα προσθέσουμε έναν Message που θα δέχεται ως Subject την λέξη Spawn.

Το animation της οπισθοδρόμησης δημιουργήθηκε με τη βοήθεια των frames. Τα ίδια θα επικαλεστούμε και στην περίπτωσή μας ώστε να εμποδίσουμε τη συνεχή εκτόξευση.

- Στον Logic Editor τουTurret προσθέτω μια Property- frame την οποία κάνω integer και δίνω αρχική τιμή 30.
- Στον Actuator- Action θέτω στο πεδίο Frame Property την μεταβλητή frame, η οποία θα κρατάει τον αριθμό των frames.
- Προσθέτω ένα sensor → Property ο οποίος θα ελέγχει αν η μεταβλητή frame έχει γίνει ίση με 30.
- Ενώνω με τον πρώτο Controller Or τον οποίο μετατρέπω σε And. Η μετατροπή είναι αναγκαία αφού θέλουμε να ισχύει ταυτόχρονα το πάτημα του Spacebar αλλά και η ισότητα του frame με τον αριθμό 30.

Ανά 30 frame επομένως, δηλαδή όσο διαρκεί και το animation θα δίνεται εντολή για δημιουργία νέας σφαίρας.

3.5 ΔΗΜΙΟΥΡΓΙΑ MONSTERS – ΕΧΘΡΩΝ

Σε αυτό το σημείο θα δημιουργήσουμε τα monsters τα οποία θα είναι οι εχθροί μας. Θα χρησιμοποιήσουμε τα κουτιά και θα τα δώσουμε τις κατάλληλες ιδιότητες.

- Shift+A →Add Mesh→ Cube Προσθέτουμε τον κύβο μας προσέχοντας μήπως χρειαστεί να το ανυψώσουμε ώστε να πατήσει πάνω στο Ground με το κουμπί G.
- Properties(καρτέλες στα δεξιά του παραθύρου) → καρτέλα Material→ Μετονομασία σε Monster
- Properties→ καρτέλα Physics→ Physics Type → Dynamic
- Properties→ καρτέλα Physics→ Collision Bounds (check) →Box ώστε να συμπεριφέρεται ως κύβος-κουτί.

Τα Monsters πρέπει φυσικά να έλκονται από το Turret μας και να προσπαθούν να το πλησιάσουν, ώστε να το εξοντώσουν. Θα χειριστούμε τα Logic Bricks του κύβου μας με τον εξής τρόπο:

- Add Sensor → Always. Θέλουμε η κίνηση να συμβαίνει συνέχεια χωρίς διακοπή
- Add Controller → Or
- Add Actuator → Edit Object, Αλλάζουμε το πεδίο Edit Object σε Track to, ώστε να ευθυγραμμιστεί ο y axis του κύβου με τον αντίστοιχο του TurretCore τον οποίο θέτουμε ως τιμή στο πεδίο Object.
- Ενώνουμε τα Bricks μεταξύ τους.

Το Monster αν και έχει στραφεί πλέον προς τον πυργίσκο μας, δεν κινείται. Για να κινηθεί πρέπει να προστεθεί ένα ακόμη Brick στους Actuators.

- Add Actuator → Motion → Loc θέτουμε στον y axis την τιμή 0.05
- Ενώνουμε το νέο Brick με τον Controller Or.

Κάθε φορά που μια σφαίρα του Turret πετυχαίνει ένα Monster, πρέπει αυτόματα να το εξαφανίζει.

- Add Sensor → Collision Αν αφήσουμε το πεδίο Property κενό τότε ο κύβος θα αλληλεπιδράσει με το πρώτο αντικείμενο με το οποίο θα έρθει σε επαφή. Στην συγκεκριμένη περίπτωση επομένως, θα τερματίσει από την αρχή κιόλας, μιας και που ήδη βρίσκεται σε επαφή με το Ground. Επομένως στο πεδίο αυτό θα πληκτρολογήσουμε το property Bullet. Σαν Bullet έχουμε ήδη ένα αντικείμενο στο δεύτερο layer. Για να ενεργοποιηθεί όμως ο αισθητήρας πρέπει να μεταβούμε το layer αυτό, να επιλέξουμε τη σφαίρα και αριστερά από το παράθυρο των Logic Bricks να ενεργοποιήσουμε το Add Game Property . Τη νέα ιδιότητα θα την ονομάσουμε Bullet.
- Add Controller → Or
- Add Actuator → Edit Object → End Object Τερματίζει το αντικείμενο.
- Ενώνουμε τα Bricks μεταξύ τους.

Αντίστοιχα με το Monster θα πρέπει να εξαφανίζεται και η σφαίρα που το πετυχαίνει. Θα παραμείνουμε επομένως στο δεύτερο layer ώστε να έχουμε τη δυνατότητα να επεξεργαστούμε τα Logic Bricks της Bullet.

- Add Sensor → Collision Στο πεδίο Property πληκτρολογούμε τη λέξη Monster. Αντίστοιχα στο πρώτο layer επιλέγουμε το Monster και προσθέτουμε την ιδιότητα Monster όπως και προηγουμένως.
- Add Controller → Or
- Add Actuator → Edit Object → End Object
- Ενώνουμε τα Bricks μεταξύ τους.

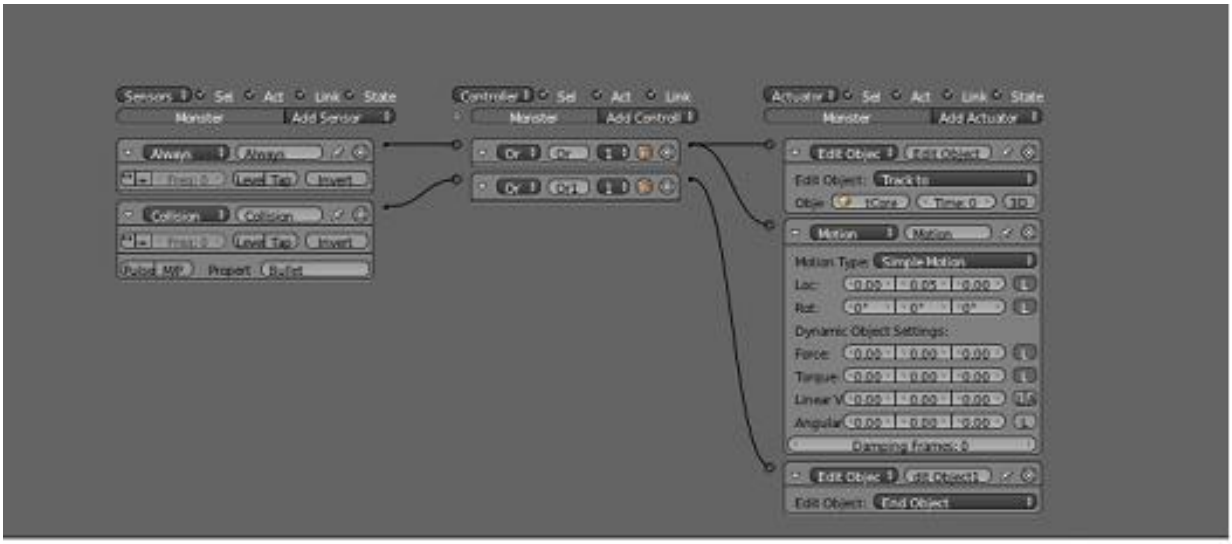

Figure 65 : **3.5.1 Logic Bricks εχθρικών κουτιών**

Για να αποκτήσει λογική το παιχνίδι, χρειάζονται παραπάνω από ένα Monster τα οποία θα έρχονται τυχαία από κάθε κατεύθυνση και με συνεχόμενο ρυθμό. Για την αναπαραγωγή των εχθρών χρησιμοποιήσαμε το ιδιαίτερο αντικείμενο Empty, το οποίο θα επιλέξουμε για ακόμη μια φορά. Προσθέτουμε λοιπόν ένα ακόμη στη σκηνή μας και το τοποθετούμε σε κάποια γωνία από την οποία θέλουμε να ξεκινάνε τα Monster. Το μετονομάζουμε από την καρτέλα Material των Properties σε MonsterSpawn. Ο κύβος μας επομένως μπορεί να μεταφερθεί στο δεύτερο layer από όπου θα τον καλούμε.

- Επιλογή του Monster
- Object → Move to Layer
- Κλικ στο layer που θέλουμε να μεταφερθεί.

Στον Logic Editor του MonsterSpawn στο πρώτο επίπεδο θα ασχοληθούμε με τα Logic Bricks τα οποία θα είναι υπεύθυνα για να καλέσουν τα Monster.

- Add Sensor→Delay - Ο συγκεκριμένος sensor ενεργοποιείται μετά από μια συγκεκριμένη χρονική καθυστέρηση, κάτι που αναζητάμε για τη συνεχή αναπαραγωγή τεράτων. Μετράει το χρόνο σε logic tics, σε 60 ανά ένα δευτερόλεπτο. Αν λοιπόν θέλουμε μια καθυστέρηση των 3 δευτερολέπτων πρέπει να ορίσουμε στο πεδίο Delay τον αριθμό 180(3*60). Αν θέλουμε να είναι επαναλαμβανόμενη η διαδικασία τότε τσεκάρουμε το πεδίο Repeat.
- Add Controller → Or
- Add Actuator → Edit Object → Add Object → Monster - Κάθε φορά που ενεργοποιείται θα προσθέτει το αντικείμενο Monster.

3.6 ΤΥΧΑΙΑ ΑΝΑΠΑΡΑΓΩΓΗ MONSTERS – ΕΧΘΡΩΝ

Ο MonsterSpawn τον οποίο δημιουργήσαμε είναι υπεύθυνος για τη συνεχή αναπαραγωγή τεράτων, αλλά μόνο από ένα συγκεκριμένο σημείο. Για να καλυφθούν οι ανάγκες του παιχνιδιού πρέπει τα τέρατα να εμφανίζονται από διάφορα σημεία της σκηνής με διαφορετικό ρυθμό. Για να δουλέψουμε αυτή τη διαδικασία θα απομονώσουμε τους Spawns, δηλαδή τα αντικείμενα Empty, στο δεύτερο Layer. Αυτό θα μας δώσει μια πιο ξεκάθαρη εικόνα της κατάστασης. Όπως ήδη ξέρουμε το δεύτερο layer είναι κατειλημμένο από τα αντικείμενα Bullet και Monster. Τα επιλέγουμε με δεξί κλικ και Shift και τα μεταφέρουμε στο Layer 6. Αντίστοιχα μεταφέρουμε από το πρώτο layer το MonsterSpawn στο δεύτερο. Στο ίδιο layer προσθέτουμε με Shift+A ένα ακόμη Empty, το οποίο μετονομάζουμε από την καρτέλα Materials σε SpawnCore. Στη συνέχεια θα επιλέξουμε το MonsterSpawn και με Shift+δεξί κλικ το SpawnCore και με CTRL+P θα τα κάνουμε Parent. Στον Logic Editor του SpawnCore θα προσθέσουμε τα εξής:

- Add Sensor → Always
- Add Controller → Or
- Add Actuator → Motion – Στο πεδίο Rot στον z axis θέτουμε την τιμή 0,3. Με αυτή τη διαδικασία κάνουμε το MonsterSpawn να περιστρέφεται κυκλικά γύρω από τον πυρήνα δημιουργώντας νέα τέρατα.
- Ενώνουμε τα Bricks μεταξύ τους.

Κάτι πολύ χρήσιμο για το παιχνίδι μας θα ήταν η δυνατότητα του SpawnCore(δηλαδή του Empty που είναι ο πυρήνας για την αναπαραγωγή εχθρών) να καθορίζει το πότε θα αναπαράγονται νέοι εχθροί. Μέχρι τώρα τη δυνατότητα αυτή είχαμε θέσει στο αντικείμενο MonsterSpawn με τον αισθητήρα(Logic Brick-Sensor) Delay. Επιλέγοντας το αντικείμενο MonsterSpawn προσθέτω τα εξής:

- Add Sensor → Delay - Θέτουμε 180 στο πεδίο delay και τσεκάρουμε το Repeat.
- Add Controller → Or
- Add Actuator → Message – Σε αυτή τη περίπτωση θα στείλουμε ένα μήνυμα στο MonsterSpawn ώστε όταν το λαμβάνει να δημιουργεί νέα Monster. Θέτουμε στο To → MonsterSpawn και στο Subject → Spawn.
- Ενώνουμε τα Bricks μεταξύ τους.

Μεταφερόμαστε στο αντικείμενο MonsterSpawn επιλέγοντάς το και διαγράφουμε τον αισθητήρα Delay. Στη θέση αυτού προσθέτουμε τον αισθητήρα Message και ως subject πληκτρολογούμαι το Spawn, ώστε κάθε φορά που δέχεται αυτό το μήνυμα να στέλνει έναν παλμό στον Actuator-Edit Object και να προσθέτει ένα Monster.

Το στοιχείο που λείπει αυτή τη στιγμή από το παιχνίδι είναι η τυχαία αναπαραγωγή στον τεράτων. Αν και δημιουργούνται συνεχώς σύμφωνα με τον ρυθμό που ορίσαμε, δεν παύει να καταφθάνουν με μια συγκεκριμένη σειρά και άρα να καθιστά τη διαδικασία αρκετά προβλέψιμη. Για την τυχαία δημιουργία Monster θα κάνουμε τις εξής αλλαγές:

- Επιλέγουμε το MonsterSpawn και με Shift+D το αντιγράφουμε 4 φορές ώστε να τοποθετηθεί στις 4 γωνίες της σκηνής μας.

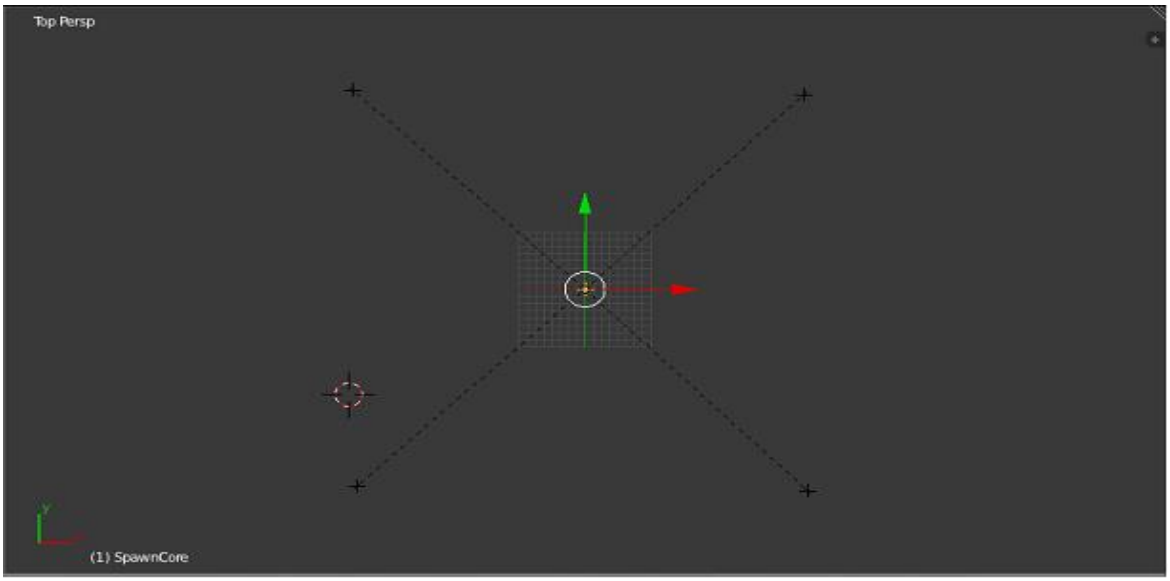

Figure 66 : **3.6.1 Διάταξη MonsterSpawn και SpawnCore**

Επιλέγουμε τον SpawnCore και αποσυνδέουμε τον Actuator-Message από τον Controller-Or. Δεν τον διαγράφουμε αλλά τον αφήνουμε για να τον επανασυνδέσουμε στη συνέχεια. Στη θέση του θα προσθέσουμε τον Actuator-Random. Ως seed θέτουμε την τιμή 1, ώστε από εκεί να ξεκινάει η τυχαία γεννήτρια. Στο πεδίο Property θα πρέπει να βάλουμε μια μεταβλητή που θα κρατάει την τιμή που θα παράγεται. Αρχικά στο αριστερά μενού Properties επιλέγουμε Add Game Property και προσθέτουμε την μεταβλητή rand την οποία μετατρέπουμε σε Integer.

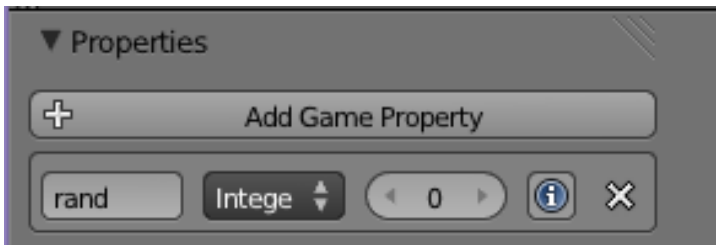

Figure 67 : **3.6.2 Property rand**

Στον Random Actuator προσθέτουμε τη μεταβλητή rand και μετατρέπουμε την Distribution Value σε Integer Uniform. Η διακύμανση των τιμών θέλουμε να είναι μεταξύ 1-4, αφού τόσα MonsterSpawn έχουμε στον χώρο, άρα στο πεδίο min προσθέτουμε την τιμή 1 και στο max την τιμή 4.

- Ενώνουμε με τον Controller Or.
- Προσθήκη τεσσάρων Property Sensor. Πεδίο Property → rand, Πεδίο Value→1,2,3,4 με τη σειρά, ώστε κάθε φορά που επιλέγεται τυχαία ένας αριθμός, να ενεργοποιούμε τον κατάλληλο MonsterSpawn.
- Προσθήκη τεσσάρων Or Controller.
- Προσθήκη τεσσάρων Message Actuator. Πεδίο Subject→ Spawn , Πεδίο To→ MonsterSpawn, MonsterSpawn.001, MonsterSpawn.002, MonsterSpawn.003 αντίστοιχα.

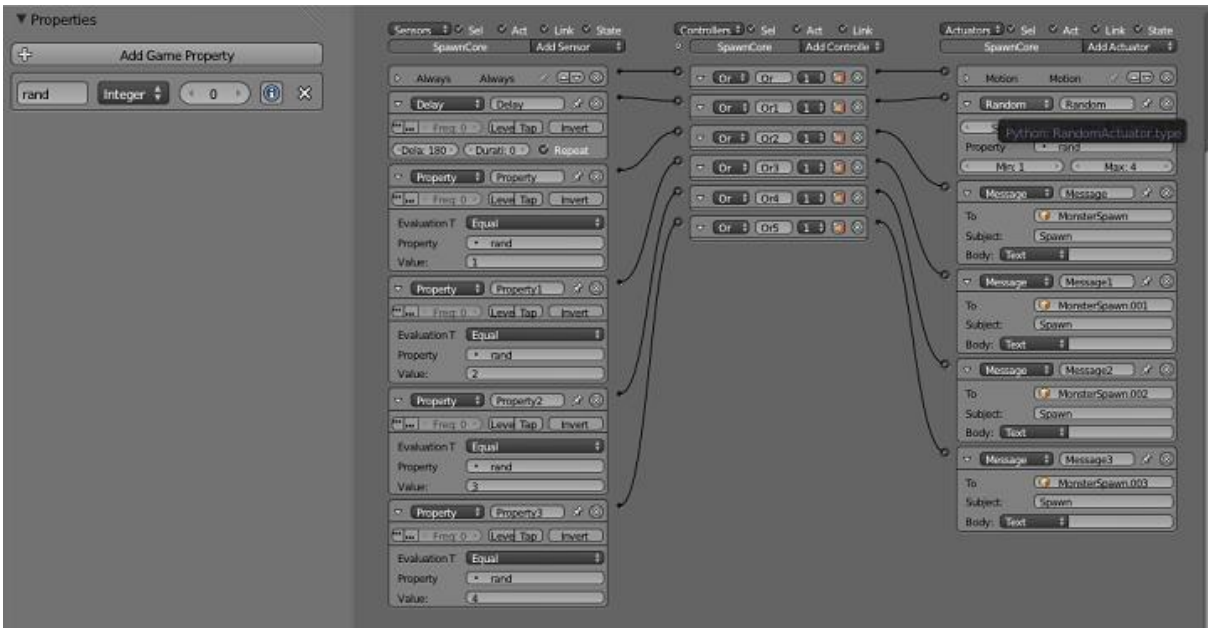

Figure 68 : 3.6.3 Ματιά στα Logic Bricks του SpawnCore

3.7 ΤΕΡΜΑΤΙΣΜΟΣ ΤΟΥ TURRET

Στην περίπτωση που κάποιο από τα Monster φτάσει το κανόνι μας χωρίς να πυροβοληθεί τότε το δεύτερο πρέπει να τερματίσει και ο παίκτης να χάσει. Για να δώσουμε κάποια διάρκεια στο παιχνίδι, ο παίκτης θα έχει 3 ευκαιρίες(health points-hp). Αν 3 Monster ακουμπήσουν το Turret τότε και μόνο τότε θα εξαφανίζεται. Μεταφερόμαστε στο πρώτο layer και με δεξί κλικ επιλέγουμε το κανόνι. Στον Logic Editor θα συμβούν οι εξής αλλαγές:

- Add Sensor → Collision – Στο πεδίο Property πληκτρολογούμε Monster, ώστε κάθε φορά που συγκρούεται με ένα να στέλνει παλμό στον Actuator.
- Add Controller→ Or
- Add Actuator→ Edit Object→ End Object

Ταυτόχρονα με το Turret θα είναι καλό να τερματίζει και η σφαίρα με την οποία συγκρούστηκε. Μεταφερόμαστε στο layer 6 και επιλέγουμε τον κύβο. Στον Logic Editor προσθέτουμε:

- Add Sensor → Collision – Στο πεδίο Property πληκτρολογούμε Turret. Επειδή ακόμη δεν υπάρχει μια τέτοια ιδιότητα, ξαναγυρνάμε στο πρώτο layer και στο αντικείμενο Turret την προσθέτουμε.
- Ενώνουμε με τον δεύτερο Controller Or ο οποίος βρίσκεται ήδη συνδεδεμένος με έναν Actuator→ End Object.

Όπως αναφέραμε και προηγουμένως ο πυργίσκος θα έχει 3 ευκαιρίες πριν τερματίσει. Θα του δώσουμε επομένως κάποια health points τα οποία θα μειώνονται κάθε φορά που εντοπίζεται μια σύγκρουση με μια σφαίρα. Για τον ορισμό τους θα χρειαστεί καταρχήν να επιλέξουμε το Turret με δεξί κλικ. Έπειτα θα ορίσουμε μια νέα property(Add Game Property) την οποία θα ονομάσουμε hp , θα μετατρέψουμε σε integer και θα δώσουμε την τιμή 3. Στον Logic Editor θα αποσυνδέσουμε τον Actuator- Edit Object από τον Controller και στη θέση του θα προσθέσουμε τον Actuator- Property. Ο συγκεκριμένος ενεργοποιητής θα χειριστεί την ιδιότητα hp. Στο πεδίο Property προσθέτουμε το hp ενώ αλλάζουμε το mode σε Add. Στο πεδίο value βάζουμε τη τιμή -1. Ενώνουμε με τον δεύτερο Controller. Με αυτόν τον τρόπο θα προστίθεται στην αρχική τιμή του hp ο αριθμός -1 κάθε φορά που θα έχουμε σύγκρουση και άρα η τιμή θα μειώνεται.

Για να έχουμε τερματισμό του αντικειμένου πρέπει να ελέγχουμε αν η τιμή των health points γίνεται 0. Οπότε προσθέτουμε ένα sensor – Property. Θέτουμε την μεταβλητή hp στο πεδίο Property και την τιμή 0 στο πεδίο Value. Προσθέτουμε έναν ακόμη Controller και ενώνουμε με το ήδη υπάρχον Edit Object που απευθύνεται στον τερματισμό του αντικειμένου

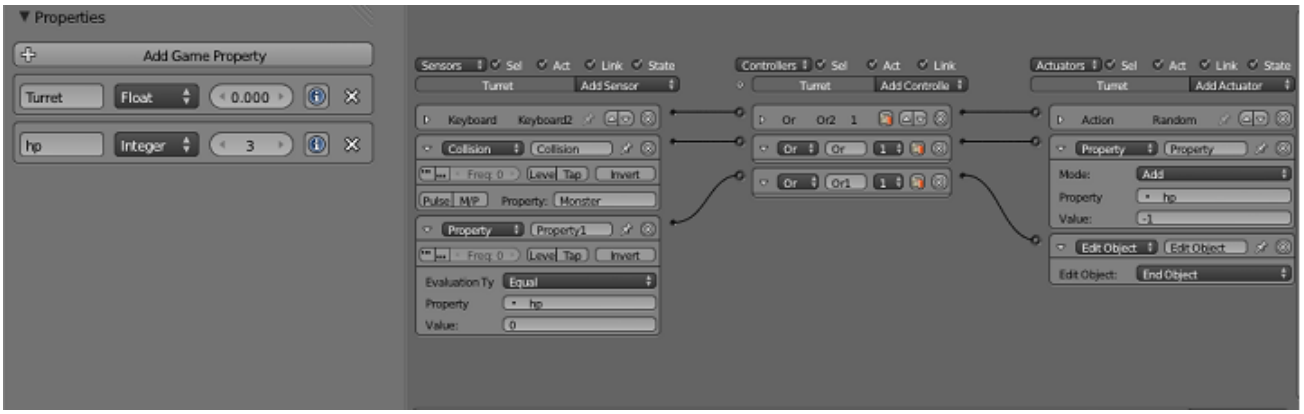

Figure 69 : **3.7.1 Properties, Logic Bricks του Turret**

Σε περίπτωση που θέλουμε να γνωρίζουμε τις αλλαγές της μεταβλητής μας την ώρα που ξεκινάμε το παιχνίδι, επιλέγουμε την ιδιότητα i στην hp Property και από το μενού Game επιλέγουμε Show Debug Properties.

3.8 EXPLOSION – ΕΚΡΗΞΗ ΠΥΡΟΒΟΛΙΣΜΟΥ

Κάθε φορά που το κανόνι εκτοξεύει μια σφαίρα, θα ήταν αρκετά ρεαλιστικό να προσθέσουμε και μια έκρηξη στη μύτη ακριβώς του κανονιού. Αυτό θα κάνει τον πυροβολισμό να φαίνεται αληθινός και θα είναι ταυτόχρονα ένα ωραίο εφέ.

- Ξεκινάμε διαλέγοντας ένα διαφορετικό σκέτο layer, προκειμένου να μην μπλέξουμε όλα τα αντικείμενα μεταξύ τους.
- Προσθέτω ένα καινούργιο plane με Shift+A.

Σε αυτό το σημείο πρέπει να ξεκαθαρίσουμε πως είναι χρήσιμο να έχουμε ήδη μια εικόνα έκρηξης αποθηκευμένη στον υπολογιστή μας, ώστε να είναι έτοιμη προς άνοιγμα και χρήση. Η εικόνα αυτή θα μπορούσε να είναι όπως η παρακάτω

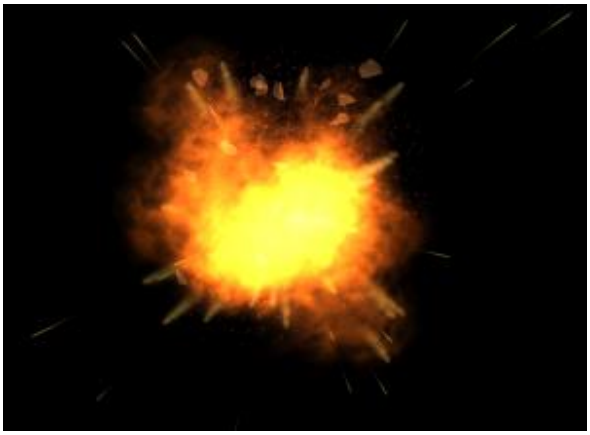

Figure 70 : **3.8.1 Σχήμα Έκρηξης**

- Στην καρτέλα Material επιλέγω new. Ενώ στην οριζόντια ενσωματωμένη καρτέλα Shading επιλέγω Shadeless.
- Στην καρτέλα Textures επιλέγω και πάλι new. Στο πεδίο Type μας ενδιαφέρει το Image or Movie. Στην οριζόντια ενσωματωμένη καρτέλα Image επιλέγω Open και εντοπίζω την εικόνα της έκρηξης που έχω στον υπολογιστή μου.
- Στην οριζόντια επίσης καρτέλα Mapping επιλέγω στο πεδίο Coordinates→UV.

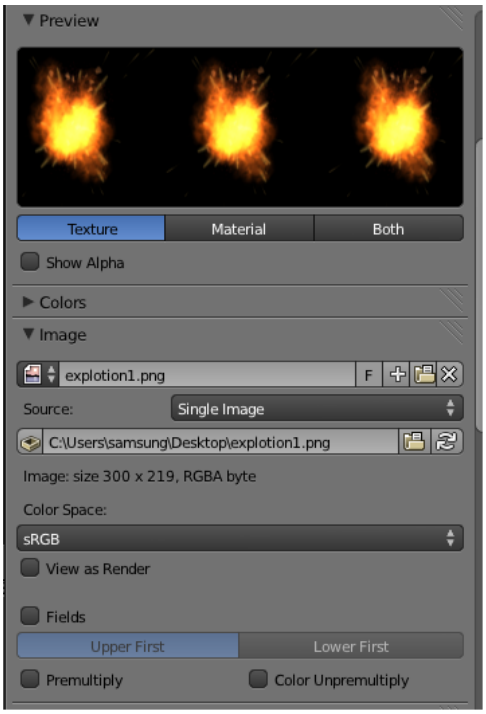

Figure 71 : 3.8.2 Προσθήκη texture στο plane της έκρηξης

- Από το default Screen layout μεταφέρομαι στο UV Editing. Θα ξεκινήσω απευθείας με την επεξεργασία μιας εικόνας ως texture.
- Πατάμε το πλήκτρο TAB και στην δεξιά οθόνη(εκεί που βρίσκεται δηλαδή το lane) πατάμε το πλήκτρο U και επιλέγουμε από το αναδυόμενο μενού το unwrap.
- Στο αριστερό παράθυρο και κάτω ακριβώς από την επιλεγμένη, κενή προς το παρόν επιφάνεια, επιλέγουμε το πεδίο δίπλα στο New και εντοπίζουμε το όνομα της εικόνας. Πρέπει να βεβαιωθούμε πως στο παράθυρο του plane βρισκόμαστε σε Texture Mode.

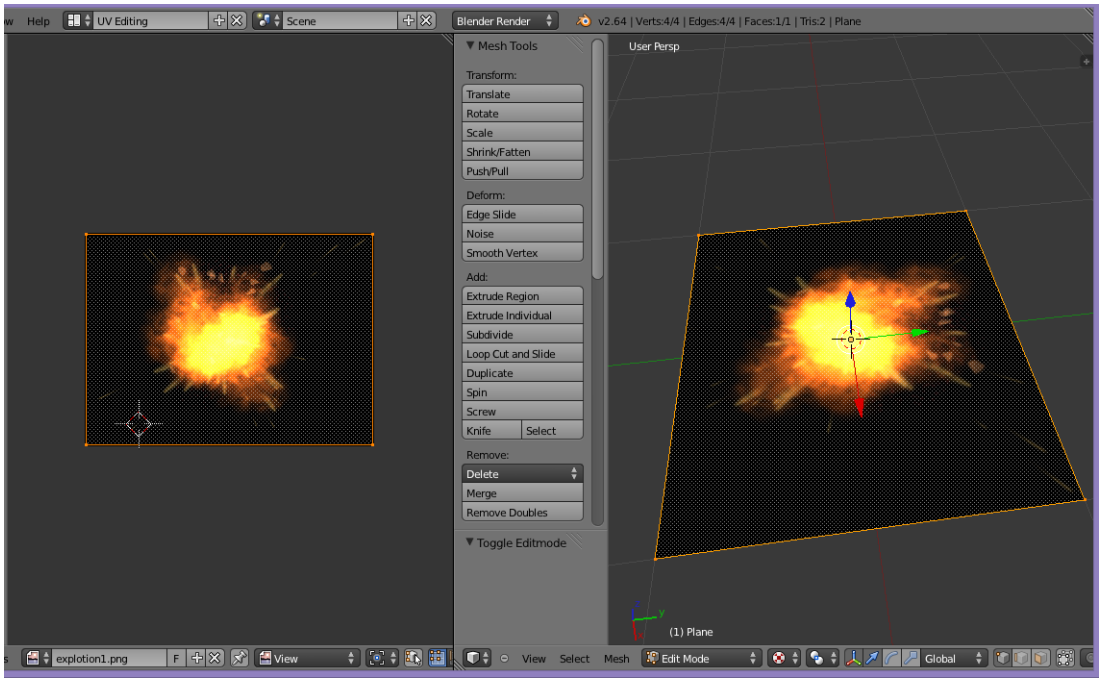

Figure 72 : 3.8.3 UV Editing

- Ξαναεπανερχόμαστε στο Defaul Screen Layout. Από την καρτέλα Material εντοπίζουμε το πεδίο Game Settings→ Alpha Bend και επιλέγουμε And. Εξαφανίζουμε λοιπόν το μαύρο background.

Figure 73 : 3.8.4 Game Settings

- Στην συνέχεια θα επαναλάβουμε κάποιες νέες κινήσεις. Με Shift+D θα πάρουμε ένα διπλότυπο του plane και απλά θα το κάνουμε Rotate γύρω από το αρχικό. Επαναλαμβάνουμε με διαφορετικές κατευθύνσεις όσες φορές θέλουμε, ώσπου να θεωρήσουμε ότι το αποτέλεσμα είναι μια έκρηξη, η οποία φαίνεται ίδια από παντού.
- Επιλέγουμε όλα τα αντικείμενα μαζί και με Ctrl+J τα ενώνουμε σε ένα κοινό.
- Στην καρτέλα Physics θέτουμε τον τύπο του νέου αντικειμένου ως Dynamic και τα Collision bounds σε Box.

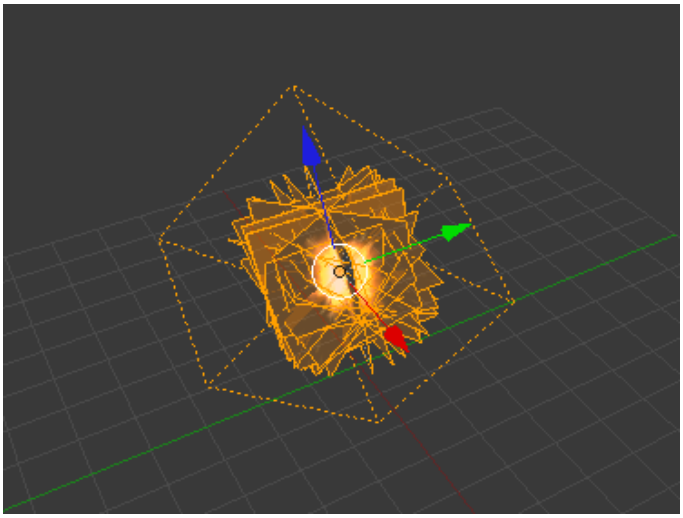

Figure 74 : **3.8.5 Join των planes**

- Πίσω στο layer ένα, δηλαδή στο αρχικό μας επιλέγουμε το Empty που είναι τοποθετημένο ακριβώς μπροστά από το κανόνι.
- Προσθέτουμε ένα ακόμη Edit Object Actuator στο Logic Panel. Στο πεδίο Object προσθέτω το αντικείμενο της έκρηξης(ανάλογα με το όνομα που το έχω δώσει), ενώ στο πεδίο Time προσθέτω τις μονάδες blender κατά τις οποίες η έκρηξη θα εμφανίζεται. Δοκιμάζω διάφορες τιμές.

ΚΕΦΑΛΑΙΟ 4

ΕΠΙΠΡΟΣΘΕΤΕΣ ΕΠΙΛΟΓΕΣ ΤΟΥ ΠΑΙΧΝΙΔΙΟΥ

Για να σταθεί ένα παιχνίδι, πρέπει εκτός από τη λογική του, ο δημιουργός του να ασχοληθεί και με extra επιλογές που θα το ολοκληρώσουν και θα το κάνουν πιο ελκυστικό στον χρήστη.

Στο κεφάλαιο αυτό θα δούμε μερικά από αυτά τα στοιχεία, απαραίτητα για ένα σωστό παιχνίδι με αρχή μέση και τέλος.

4.1 ΔΗΜΙΟΥΡΓΙΑ ΑΥΤΟΝΟΜΟΥ ΠΑΙΧΝΙΔΙΟΥ

Μεταφερόμαστε στον User Preferences Editor.

Διαλέγουμε την καρτέλα Addons.

Από την λίστα Categories επιλέγουμε το Game Engine.

Τσεκάρουμε το κουτάκι που αντιστοιχεί στο Game Engine: Save As Game Engine Runtime.

Από το info Editor επιλέγουμε File και έπειτα Export → Save as Game Engine Runtime.

Δίνουμε ένα όνομα στην εφαρμογή μας και αποθηκεύουμε.

Figure 75 : 4.1.1 User Preferences - Addons

4.2 ΔΗΜΙΟΥΡΓΙΑ ΔΥΝΑΜΙΚΟΥ ΚΕΙΜΕΝΟΥ

Σε ένα παιχνίδι όπως το Turret Defense μια καλή τακτική είναι η ενσωμάτωση κειμένου που αφορά τα health points, τον χρόνο αλλά και τις νίκες που έχουμε καταφέρει μέχρι την συγκεκριμένη χρονική στιγμή. Έχουμε ήδη κάποιες από τις μεταβλητές που βοηθάνε σε αυτή τη δουλειά στο παιχνίδι μας. Το μόνο πρόβλημα είναι πως χρειαζόμαστε κείμενο Realtime. Δηλαδή να εμφανίζεται καθώς το παιχνίδι τρέχει και να μπορεί να διαμορφωθεί ανάλογα με την πορεία του παιχνιδιού.

Στο blender η διαδικασία αυτή δεν είναι τόσο απλή, αλλά δεν χρειάζονται παρά μόνο μερικά βήματα για την υλοποίηση της. Αρχικά πρέπει να αναφέρουμε ότι για την τοποθέτηση κειμένου πρέπει να χρησιμοποιήσουμε μια εικόνα γραμματοσειράς(font image) όπως η παρακάτω:

Figure 76 : 4.2.1 Font Image

Αν θέλουμε να δημιουργήσουμε τη δικιά μας γραμματοσειρά από τα αρχεία .ttf τότε χρησιμοποιούμε το πρόγραμμα ftblender.exe . Το κατεβάζουμε από το διαδίκτυο και το αποσυμπιέζουμε στον φάκελό του.

Το πρόγραμμα αυτό δημιουργεί γραμματοσειρές bitmap. Βρίσκουμε την γραμματοσειρά που θέλουμε και την αντιγράφουμε μέσα στον ίδιο φάκελο. Θα παρατηρήσουμε ότι μέσα στον φάκελο βρίσκονται δύο αρχεία. Ένα .blend και ένα .exe. Μετά την επικόλληση ανοίγουμε το .blend με διπλό κλικ. Ανοίγει ένα παράθυρο με έναν κώδικα Python. Πατάμε Alt+P για να τρέξουμε το script και ένα καινούργιο παράθυρο θα ανοίξει με την εικόνα της γραμματοσειράς. Πατάμε F2 και αποθηκεύουμε την εικόνα ως αρχείο .tga . Το F1 δίνει τη λίστα με τις εντολές.

Συνεχίζουμε γυρνώντας πίσω στο παιχνίδι. Βρισκόμαστε σε Camera Perspective. Δημιουργούμε μία νέα σκηνή από τη γραμμή εργασιών στην κορυφή του προγράμματος και την ονομάζουμε με ένα άλλο όνομα ώστε να ξεχωρίζει από την αρχική. Με B και σύρσιμο του ποντικιού επιλέγουμε τα πάντα εκτός από το παράθυρο που υποθετικά είναι η κάμερά μας και τα διαγράφουμε με Delete.

Επιλέγουμε και την κάμερα και πατάμε τους συνδυασμούς Alt+G, Alt+R για να αρχικοποιήσουμε τη θέση και τον προσανατολισμό. Πατάμε 1 και βρισκόμαστε Front Ortho και κάτω από την κάμερα τοποθετούμε ένα νέο plane, το οποίο θα χρησιμεύσει για την διαμόρφωση του κειμένου μας. Γυρνάμε σε Camera Persp για να έχουμε συνολική εικόνα και μεταβιβαζόμαστε σε Texture Mode.

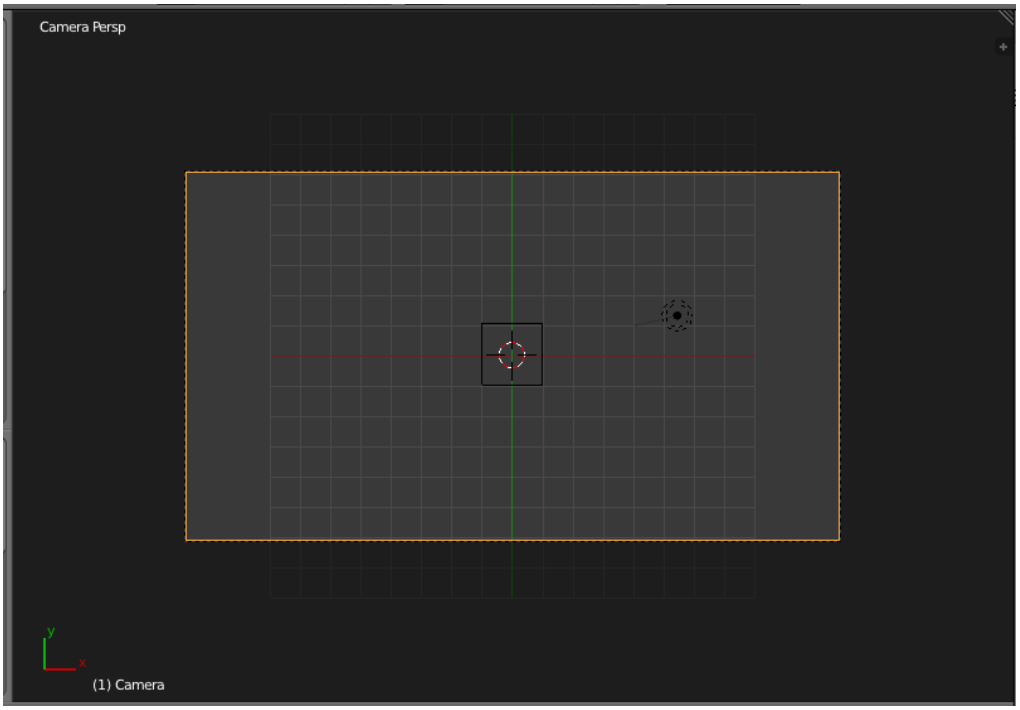

Figure 77 : **4.2.2 Camera Persp**

Από το μενού στα δεξιά περνάμε στην καρτέλα Material και προσθέτουμε ένα νέο. Το κάνουμε μαύρο από το Diffuse και shadeless από το Shading. Περνάμε στην καρτέλα Texture και αλλάζουμε το Type σε Image or Movie. Στη συνέχεια ανοίγουμε το αρχείο tga που αποθηκεύσαμε προηγουμένως. Στο Mapping αλλάζουμε το Generated σε UV.

Για να τοποθετήσουμε την εικόνα με τη γραμματοσειρά πρέπει από Default layout να περάσουμε σε UV Editing. Σιγουρευόμαστε ταυτόχρονα πως η επιλεγμένη σκηνή είναι η καινούργια. Παραμένοντας σε Texture περνάμε σε Edit Mode και με πάτημα του U επιλέγουμε unwrap.

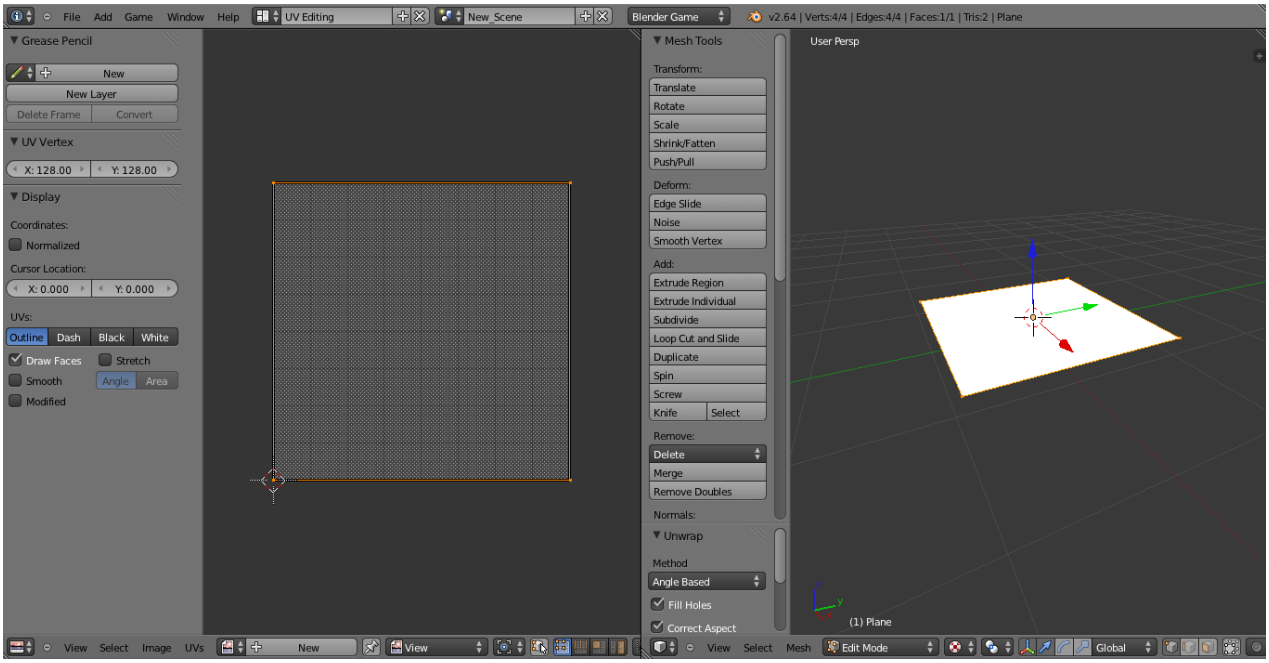

Figure 78 : 4.2.3 UV Editing

Στο αριστερό παράθυρο συναντάμε τώρα την επιλεγμένη UV περιοχή. Από την κάτω γραμμή εργαλείων και δίπλα στο κουμπί OPEN ανοίγουμε την tga εικόνα που έχουμε εισάγει.

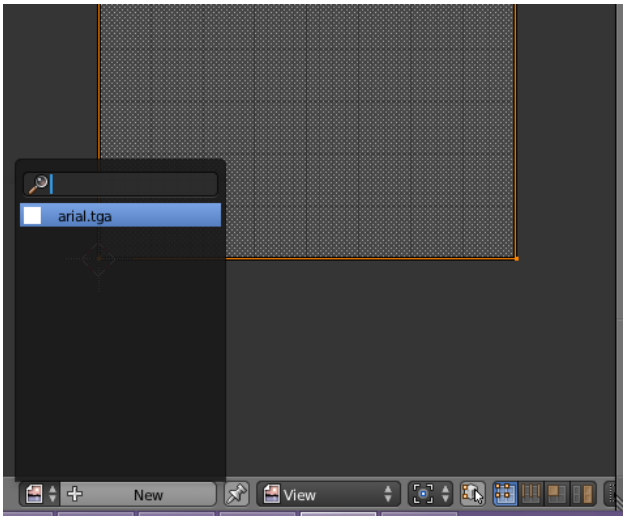

Figure 79 : 4.2.4 New→ Επιλογή εικόνας

Με το πλήκτρο S μικραίνουμε την επιλεγμένη περιοχή και την τοποθετούμε πάνω από το σύμβολο @. Κάνουμε τις απαραίτητες αλλαγές ώστε να χωρέσει ακριβώς.

Έπειτα γυρίζουμε πίσω στην Default σκηνή μας και με επιλεγμένο το plane επιλέγουμε την αριστερή καρτέλα, Material. Κατεβαίνοντας προς τα κάτω συναντάμε την οριζόντια καρτέλα Game Settings και θέτουμε με τικ το Text. Γυρίζουμε σε Logic Editor και προσθέτουμε την ιδιότητα που θα ονομάσουμε Text η οποία θα είναι τύπου Time. Ως πρώτο παράδειγμα θα φτιάξουμε έναν μετρητή.

Αν τρέξουμε το παιχνίδι το πιο πιθανό είναι ο μετρητής μας να είναι ανάποδα, οπότε με R κάνουμε περιστροφή του plane και ξαναδοκιμάζουμε. Αν δεν θέλουμε το μαύρο φόντο τότε από την καρτέλα Game Settings και πάλι αλλάζουμε το Opaque σε Add.

Σε αυτό το σημείο πρέπει να ενσωματώσουμε την σκηνή αυτή στην κεντρική. Επομένως γυρνάμε στην αρχική μας και επιλέγουμε την κάμερα CameraTPS. Χειριζόμαστε για άλλη μια φορά τα logic bricks.

- Sensor→ Always
- Controller → Or
- Actuator → Scene → Add Overlay Scene→ Προσθήκη του ονόματος της νέας σκηνής.

4.3 ΔΗΜΙΟΥΡΓΙΑ ΠΕΡΙΒΑΛΛΟΝΤΟΣ ΓΥΡΩ ΑΠΟ ΤΟ ΠΑΙΧΝΙΔΙ

Το παιχνίδι θα μπορούσε να σταθεί και μόνο του αλλάζοντας τις ιδιότητες του περιβάλλοντος από την καρτέλα World, προσθέτοντας χρώμα για τον ουρανό, ίσως ένα είδος ομίχλης κτλ. Αντί για αυτό για προσθέσουμε ένα ολόκληρο φυσικό τοπίο με έναν απλό αλλά λειτουργικό τρόπο.

Στην ήδη υπάρχουσα σκηνή προσθέτουμε με Shift+A μια UV Sphere. Την σφαίρα αυτή θα την μεγαλώσουμε και θέλουμε μέσα της να υπάρχουν όλα τα αντικείμενα ή αλλιώς ολόκληρη η σκηνή. Πατάμε S και έπειτα το νούμερο 5 ώστε να μεγαλώσει επί πέντε φορές το μέγεθός της. Επαναλαμβάνουμε έως ότου βρεθεί ολόκληρο το plane μέσα στη σφαίρα.

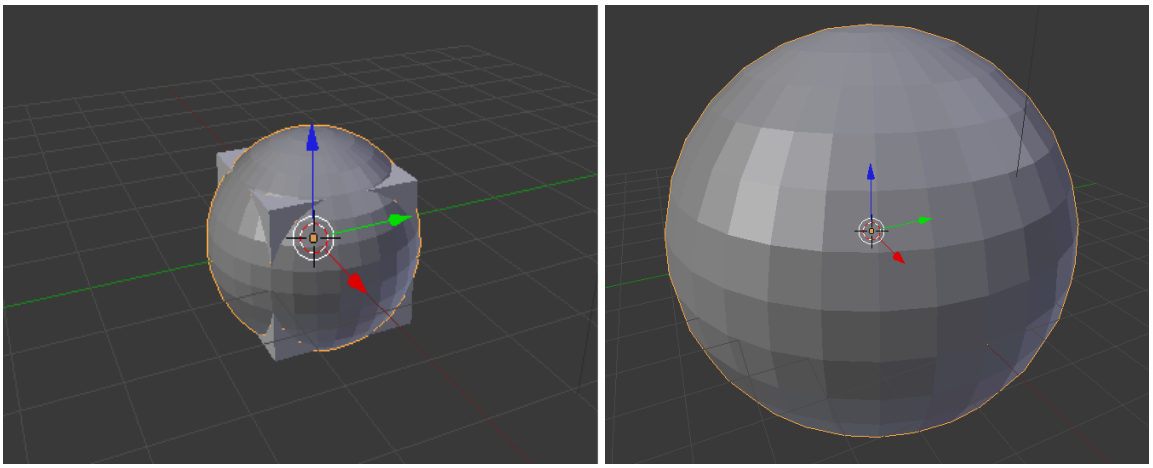

Figure 80 : **4.3.1 Μεγέθυνση σφαίρας**

Με επιλεγμένη τη σφαίρα επιλέγω την καρτέλα Render στα δεξιά της οθόνης. Από το πεδίο Shading θα μεταβώ σε GLSL. Η κίνηση αυτή να κάνει ενδεχομένως όλα τα αντικείμενο θολά, αφού ο φωτισμός επηρεάζεται από τη σκίαση. Επιλέγω τα υπόλοιπα αντικείμενα εκτός της σφαίρας που τα περιβάλλει και αλλάζω και πάλι το Shading σε Multitexture.

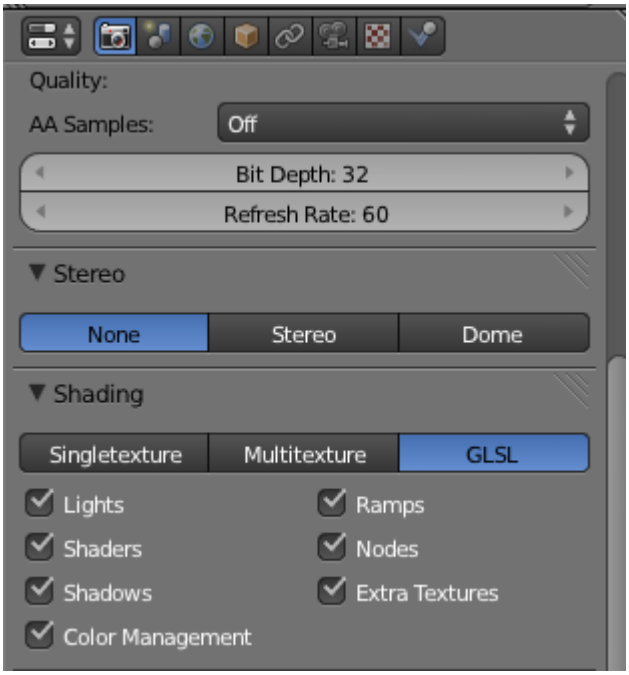

Figure 81 : 4.3.2 Καρτέλα Render→ Shading

Για τη δημιουργία του περιβάλλοντος θα χρησιμοποιήσω ως texture κάποια εικόνα που θα έχω ήδη κατεβάσει και αποθηκεύσει στον υπολογιστή μου. Πρώτο βήμα η online αναζήτηση εικόνων που μπορούν να προσαρμοστούν σε σφαίρα. Λέξεις για την επιτυχή εύρεσή τους είναι: sky dome, spherical map και άλλες. Σε αυτό το σημείο θέτουμε την σκηνή σε Texture Shading αν δεν βρίσκεται ήδη και Edit Mode για την εισαγωγή της εικόνας. Από το Material tab επιλέγω new και κάνω κλικ την επιλογή Shadeless .

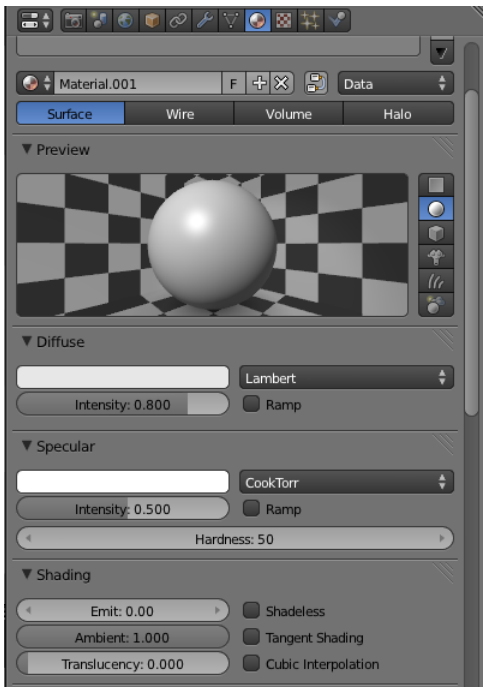

Figure 82 : 4.3.3 Καρτέλα Material

Από το Texture tab επιλέγω new και ως τύπο Image or Movie. Με το Open ανοίγω την εικόνα που έχω αποθηκεύσει στον υπολογιστή μου, ενώ από το πεδίο Mapping επιλέγω Coordinates→ UV. Η εικόνα εμφανίζεται στην εσωτερική επιφάνεια της σφαίρας αλλά παρατηρούμε πως είναι τοποθετημένη με λάθος κατεύθυνση.

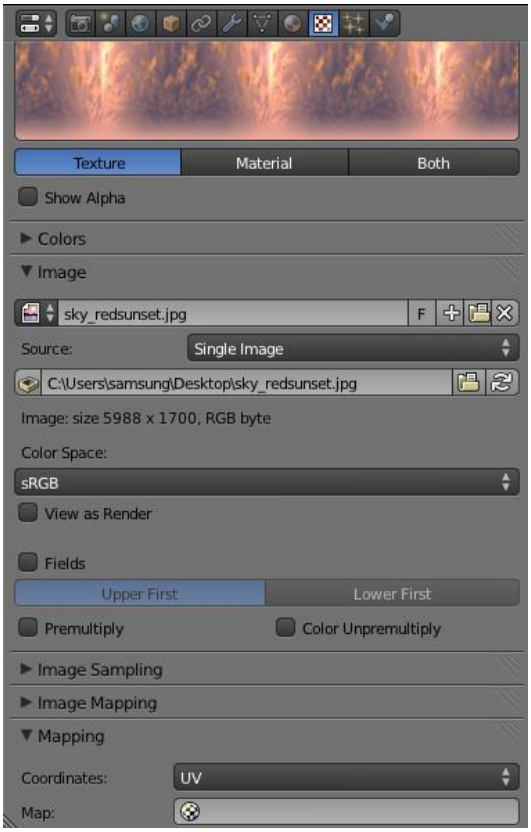

Figure 83 : 4.3.4 Καρτέλα Texture

Στην αριστερή μεριά της οθόνης βρίσκονται τα Mesh Tools. Θα χειριστούμε κάποια εργαλεία για τη σωστή τοποθέτηση της εικόνας. Στο πεδίο UV Mapping υπάρχει η επιλογή Unwrap. Με το πάτημα σε αυτήν εμφανίζεται ένα μενού από το οποίο θα επιλέξουμε Sphere Projection ώστε να προσαρμοστεί στη σφαίρα. Αμέσως εμφανίζεται ένα νέο μενού ακριβώς από κάτω στο οποίο αλλάζουμε το πεδίο Direction σε Align to Object.

Figure 84 : 4.3.5 Mesh Tools με την επιλογή TAB ενός αντικειμένου

Αν τρέξουμε το παιχνίδι θα παρατηρήσουμε πως το background δεν φαίνεται. Αντί αυτού έχουμε μια γκρίζα οθόνη. Αυτό συμβαίνει γιατί τα faces του κάθε κουτιού στο Edit Mode είναι ρυθμισμένα κοιτάνε προς την εξωτερική επιφάνεια ενός αντικειμένου, με αποτέλεσμα το texture που προσθέτουμε να είναι ορατό μόνο από την εξωτερική μεριά. Βγαίνοντας έξω από τη σφαίρα μεταφερόμαστε και πάλι με tab σε Edit Mode. Στην δεξιά μεριά της οθόνης 3d View υπάρχει κλειστή ή ανοιχτή μια λίστα εργαλείων για τα αντικείμενα. Αν είναι κλειστή τότε την ανοίγουμε από το σύμβολο + στο δεξί πάνω μέρος. Στο πεδίο Mesh Display επιλέγουμε από το Normals: το εικονίδιο που αντιστοιχεί στην επιλογή Faces. Επάνω στο επιλεγμένο αντικείμενο(σφαίρα) εμφανίζονται κάποιες μπλε γραμμές, οι οποίες υποδηλώνουν τις επιφάνειες που είναι ορατές από τον χρήστη.

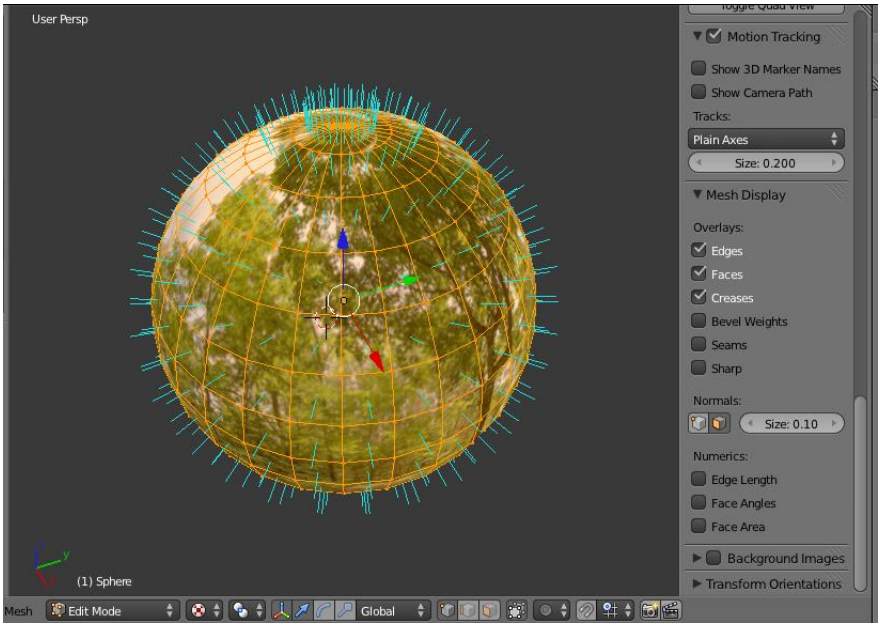

Figure 85 : 4.3.6 Faces από την εξωτερική μεριά

Προς το παρόν ορατή είναι η εξωτερική πλευρά της σφαίρας. Για να κάνουμε ορατή την εσωτερική πλευρά ασχολούμαστε και πάλι με την αριστερή καρτέλα Mesh Tools και επιλέγουμε αυτή τη φορά το πεδίο Normals: → Flip Direction. Οι μπλε γραμμές είναι ορατές μόνο από την εσωτερική επιφάνεια. Αν τρέξουμε το παιχνίδι θα διαπιστώσουμε πως το πρόβλημα έχει λυθεί.

4.4 ΔΗΜΙΟΥΡΓΙΑ ΟΘΟΝΗΣ ΕΚΚΙΝΗΣΗΣ

Για την ομαλή ένταξη του παίκτη στο παιχνίδι μας, θα χρησιμοποιήσουμε μια οθόνη εκκίνησης(Splash Screen). Η οθόνη αυτή θα περιλαμβάνει τα κουμπιά έναρξης και τερματισμού του παιχνιδιού, για την ομαλή εκκίνηση του παιχνιδιού ή ακύρωση της ενέργειας.

Η οθόνη εκκίνησης θα αποτελεί μια καινούργια σκηνή(scene) στο παιχνίδι, άρα ξεκινάμε με File→ New. Η σκηνή μας θα είναι Blender Game. Η σύνδεση με το παιχνίδι μας θα γίνει στο τελευταίο στάδιο. Αρχικά διαγράφουμε από τη νέα σκηνή τον κύβο και το φως. Προσθέτουμε με Shift+A ένα plane και με το πλήκτρο S το μεγαλώνουμε λίγο προς τον άξονα y και έπειτα προς τον άξονα x. Καλό είναι σε αυτό το σημείο να μεταβούμε με το πλήκτρο 7 σε Top Perspective View. Με τον συνδυασμό των πλήκτρων Ctrl+Alt+0 τοποθετούμε την κάμερα ακριβώς πάνω από το Plane και ταυτόχρονα έχουμε θέα μέσω αυτής. Με επιλεγμένη την κάμερα εντοπίζουμε την καρτέλα Object Data και θέτουμε τον φακό σε Orthographic. Μας ενδιαφέρει το plane να χωρέσει ακριβώς στην κάμερα μας και να καλύπτει όλη τη θέα. Με S το μεγαλώνουμε στον άξονα y και x έως ότου καλύψει πλήρως την επιφάνεια.

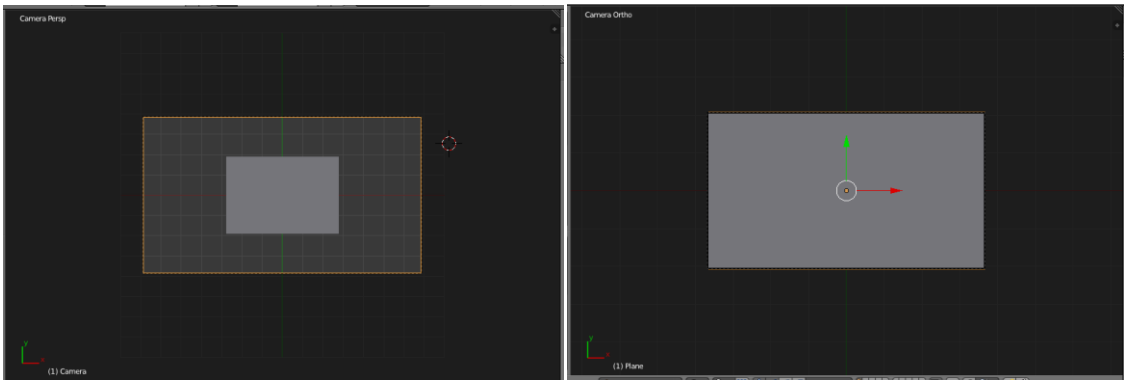

Figure 86 : 4.4.1 Cam Persp στα δεξιά, Cam Ortho στα αριστερά

Η οθόνη εκκίνησης θα πρέπει σε αυτό το σημείο να έχει ένα texture, δηλαδή ένα φόντο στο οποίο θα εφαρμοστούν οι επιλογές για τον χρήστη. Μπορούμε επομένως να έχουμε ήδη κατεβάσει και αποθηκεύσει στον υπολογιστή μας μια εικόνα κατάλληλη για το παιχνίδι. Περνάμε σε Texture Mode και από την καρτέλα Texture αλλάζουμε το Type σε Image or Movie. Στη συνέχεια ανοίγουμε την εικόνα που αποθηκεύσαμε στον υπολογιστή. Στο Mapping αλλάζουμε το Generated σε UV.

Για να τοποθετηθεί η εικόνα πρέπει από Default layout να περάσουμε σε UV Editing. Σιγουρευόμαστε ταυτόχρονα πως η επιλεγμένη σκηνή είναι η καινούργια. Παραμένοντας σε Texture περνάμε σε Edit Mode και με πάτημα του U επιλέγουμε unwrap. Στο αριστερό παράθυρο συναντάμε τώρα την επιλεγμένη UV περιοχή. Από την κάτω γραμμή εργαλείων και δίπλα στο κουμπί OPEN ανοίγουμε την εικόνα που έχουμε εισάγει.

Έχοντας τοποθετήσει την εικόνα που θα χρησιμοποιήσουμε για φόντο, μένει η προσθήκη του κειμένου που θα χρησιμοποιηθεί ως κουμπί-πλήκτρο, για τη μετάβαση στο παιχνίδι, τον τερματισμό, ή την οθόνη Βοήθειας. Με Shift+A προσθέτω ένα αντικείμενο Text. Με το πλήκτρο S το μικραίνω και το τοποθετώ στη μεριά που του αρμόζει. Με το πλήκτρο G το ανυψώνουμε στον άξονα z, διότι στη συνέχεια θα τοποθετηθεί από κάτω ένα plane, το οποίο θα αναλάβει τις διαδικασίες λογικής.

Πολύ εύκολα μπορούμε πάνω στο κείμενο να προσαρμόσουμε το δικό μας Font.

- Καρτέλα Object Data
- Font→ Επιλογή τύπου γραμμάτων(Regular, Bold, Italic, Bold & Italic)
- Κάνουμε κλικ στο κουμπί Load και άνοιγμα του αρχείου font από τη λίστα, εφόσον μεταβούμε στον φάκελο Fonts.

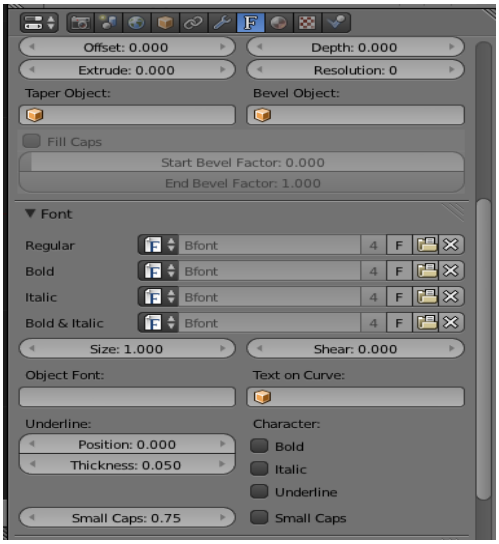

Figure 87 : 4.4.2 Εισαγωγή γραμματοσειράς στο αντικείμενο Text

Για να αλλάξουμε το κείμενο και να πληκτρολογήσουμε αυτό της επιλογής μας, θα μεταβούμε σε Edit Mode και με Backspace θα σβήσουμε αυτό που υπάρχει για να προσθέσουμε το νέο. Στο κείμενο αυτό μπορεί να προστεθεί χρώμα από την καρτέλα Material.

Στο σημείο αυτό θα με Shift+A θα προσθέσουμε ένα νέο Plane το οποίο θα χειρίζεται τη λογική διαδικασία. Θα φροντίσουμε να βρίσκεται κάτω από το κείμενο, αλλά πάνω από την εικόνα φόντου. Έπειτα θα το προσαρμόσουμε στο μέγεθος του κειμένου. Για προσθήκη περαιτέρω επιλογών στο μενού, απλά θα αντιγράψουμε το κείμενο μαζί με το plane και θα τα τοποθετήσουμε σε κάποιο άλλο σημείο, ενώ με Edit Object Mode θα πληκτρολογήσουμε το καινούργιο κείμενο. Για κάθε μία από τις επιλογές του μενού θα θέσουμε το plane ως γονέα του κειμένου.

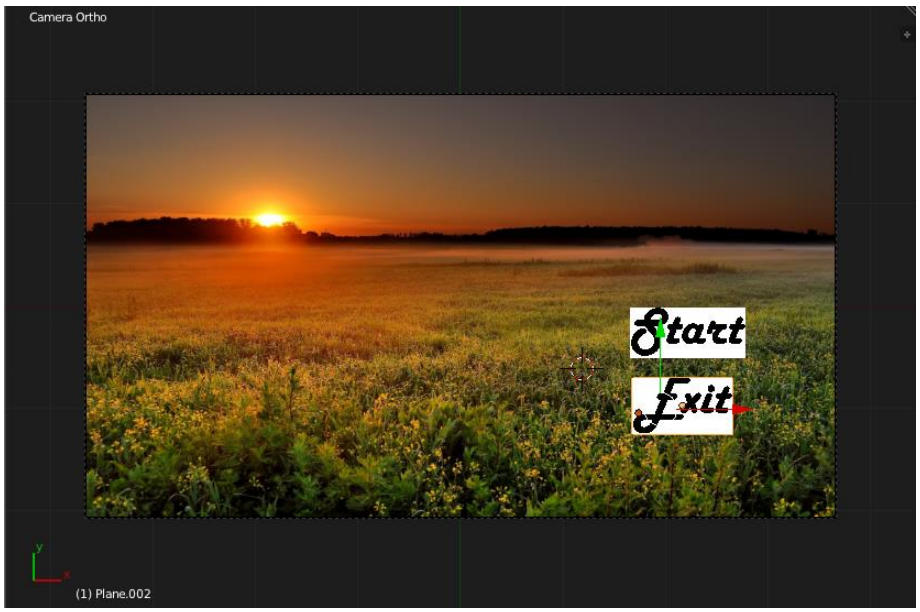

Figure 88 : 4.4.3 Αντικείμενα Plane πίσω από αντικείμενα Text

Για κάθε ένα από τα κείμενα, θα τα επιλέξουμε και στην καρτέλα Physics θα θέσουμε τον τύπο No Collision, αφού δεν θέλουμε κάποια αλληλεπίδραση με το περιβάλλον. Αντίθετα τα plane θα τα θέσουμε Actors αλλά και Invisible, αφού δεν θέλουμε να είναι εμφανή καθώς τρέχει το παιχνίδι.

4.4.1 ANIMATION MOUSE OVER

Ένα εφέ που θα διευκολύνει τον παίκτη είναι αυτό στο οποίο όταν το ποντίκι περνάει πάνω από τα κουμπιά των επιλογών, αυτά θα αυξομειώνονται. Θα χρησιμοποιήσουμε το Animation layout, άρα θα πρέπει να μεταβούμε σε αυτό από το Default που βρίσκεται η σκηνή. Με δεξί κλικ και Shift επιλέγουμε και τα δύο plane και με το πλήκτρο I κλειδώνουμε το τρέχον scaling. Στο frame 1 επομένως έχουμε την τρέχουσα κατάσταση.

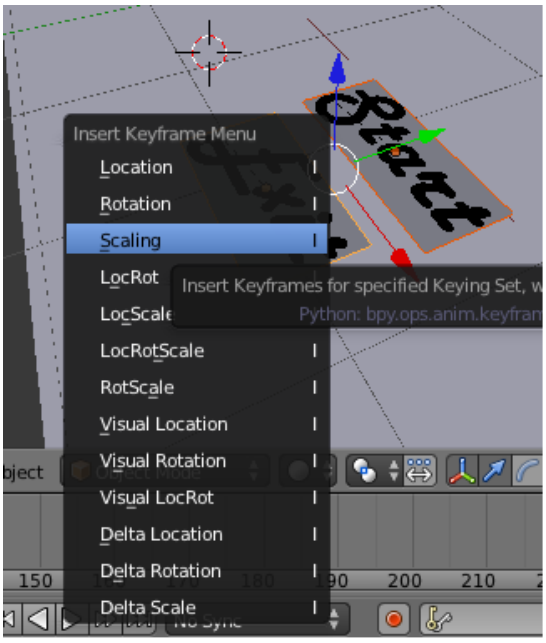

Figure 89 : **4.4.1.1 πλήκτρο I, επιλογή Scaling**

Από το παράθυρο αριστερά, δηλαδή το παράθυρο του Animation, μετακινούμε την πράσινη μπάρα στο frame 2. Επιλέγουμε ξανά τα δύο plane και με το πλήκτρο S τα μεγαλώνουμε. Έπειτα με το πλήκτρο I κλειδώνουμε το scaling για ακόμη μια φορά.

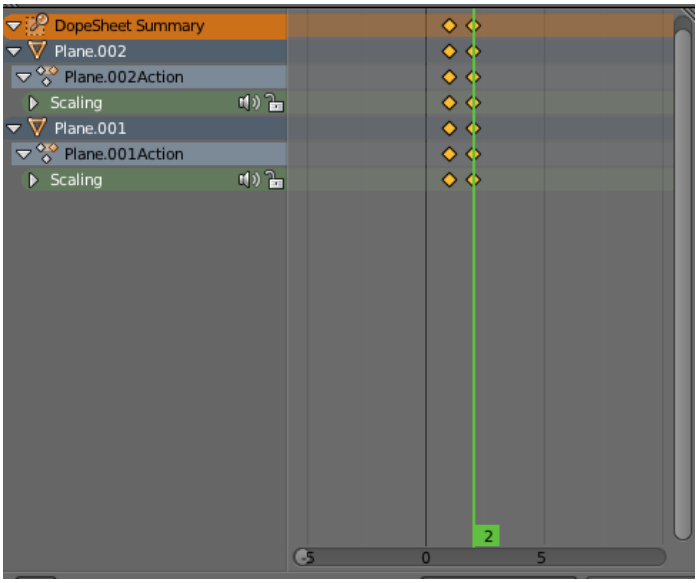

Figure 90 : 4.4.1.2 Dopesheet, keyframes

Επιστρέφουμε στο Default Screen Layout και είμαστε έτοιμοι να χειριστούμε τα Logic Bricks. Χρησιμοποιώντας λοιπόν τον Logic Editor και με επιλεγμένο το plane του κειμένου που αφορά την εκκίνηση του παιχνιδιού, κάνουμε τις εξής ενέργειες:

- Add Sensor→ Mouse. Στο πεδίο Mouse Event διαλέγουμε από τη λίστα το Mouse Over, αφού κάθε φορά που το ποντίκι περνάει από πάνω, θέλουμε να εκτελείται το animation που θέσαμε.
- Add Controller→ And
- Add Actuator→ Action. Επιλέγουμε το animation, τσεκάρουμε το πεδίο Continue και ως Start Frame βάζουμε το 1, ενώ ως End Frame το 2. Με αυτόν τον τρόπο το κείμενο θα εκτελεί το μεγάλωμα έτσι ακριβώς όπως το θέσαμε για αυτά τα δυο frame.
- Ενώνουμε Sensor με Controller και Actuator.
- Add Controller→ Nand
- Add Actuator→ Action. Επιλέγουμε το animation, τσεκάρουμε το πεδίο Continue και ως Start Frame βάζουμε το 2, ενώ ως End Frame το 1. Με αυτόν τον τρόπο εκτελεί την αντίθετη κίνηση, δηλαδή το κείμενο επανέρχεται στην αρχική του κατάσταση.
- Ενώνουμε τον αρχικό Sensor και με τον νέο Controller και με τον νέο Actuator. Όταν λοιπόν το ποντίκι ΔΕΝ θα είναι πάνω από το κείμενο, αυτό θα επανέρχεται στο αρχικό του μέγεθος.
- Add Sensor→ Mouse. Στο Mouse Event επιλέγουμε το Left Button, αφού θέλουμε με το κλικ του ποντικιού να ξεκινάει το παιχνίδι Turret Defence.
- Add Controller→ And
- Add Actuator→ Game. Πεδίο Game→ Start Game From File. Πεδίο File→ Επιλογή του παιχνιδιού.

- Ενώνουμε και τους δύο Αισθητήρες με τον καινούργιο Controller And και έπειτα τον And με τον νέο Ενεργοποιητή. Επομένως όταν το ποντίκι περνάει πάνω από το κείμενο και ταυτόχρονα γίνετε κλικ, θα φορτώνεται το παιχνίδι.

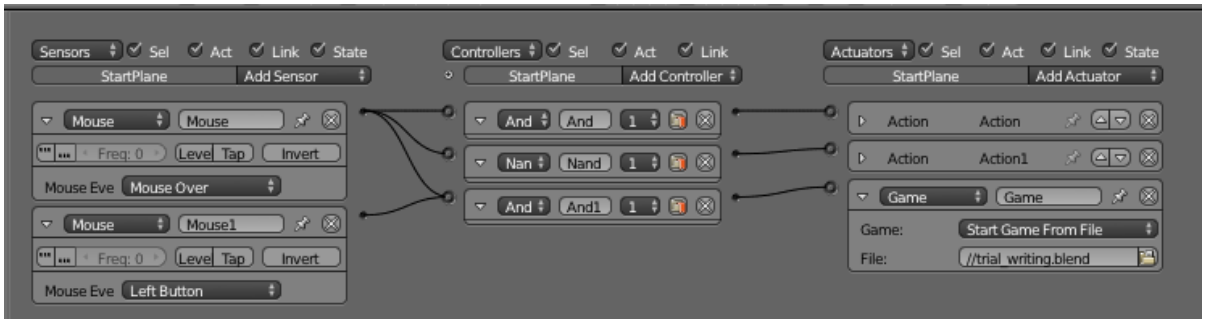

Figure 91 : **4.4.1.3 LogicBricks για τα αντικείμενα Text**

Η διαδικασία θα επαναληφθεί για τα υπόλοιπα κουμπιά του μενού της οθόνης εκκίνησης, πραγματοποιώντας τις κατάλληλες αλλαγές στον Ενεργοποιητή Game.

Για να είμαστε όμως σε θέση να επιλέξουμε κάποιο από τα κουμπιά, κατά τη διάρκεια του Runtime, θα πρέπει το ποντίκι να εμφανίζεται στην οθόνη εκκίνησης. Για να το πετύχουμε αυτό θα γράψουμε δυο γραμμές κώδικα Python στον Text Editor. Στην θέση του Logic Editor επιλέγω τον Text Editor και πληκτρολογώ:

import Rasterizer

Rasterizer.showMouse(1)

Το ποντίκι πλέον θα είναι ενεργό κατά τη διάρκεια που η σκηνή θα τρέχει.

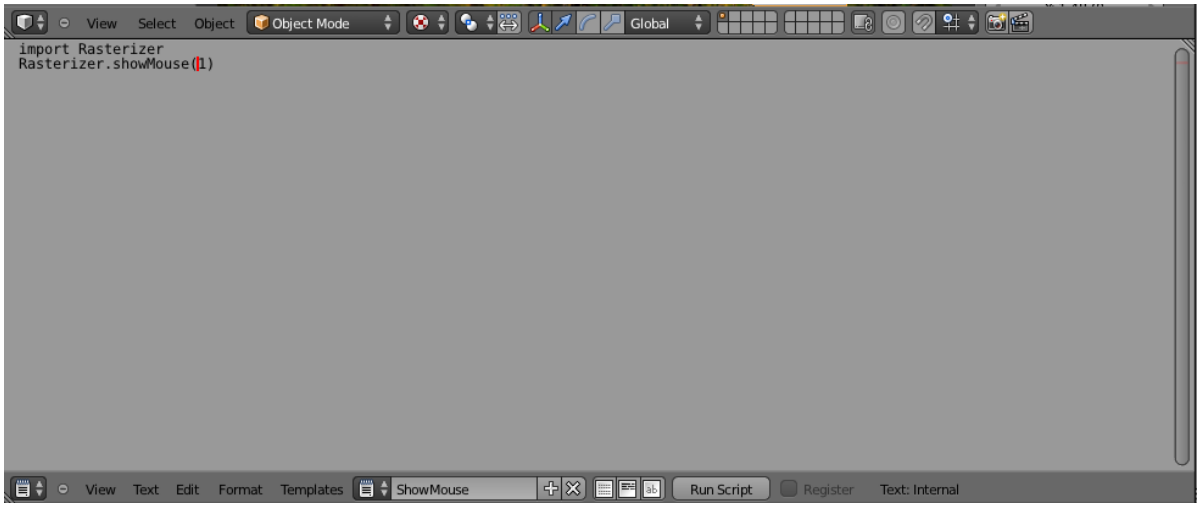

Figure 92 : **4.4.1.4 Text Editor**

4.5 Export-Import ΑΝΤΙΚΕΙΜΕΝΑ ΑΠΟ ΑΛΛΕΣ ΣΚΗΝΕΣ

Για την εξαγωγή και εισαγωγή αντικειμένων από μια σκηνή στην άλλη υπάρχουν έτοιμες στο Blender οι επιλογές Export και Import. Όταν θέλω επομένως να μεταφέρω αντικείμενα από τη μια σκηνή στην άλλη επιλέγω από το μενού File την επιλογή Export→ Wavefront (.obj). Η επιλογή αυτή χρησιμοποιείται για την εξαγωγή αλλά και εισαγωγή καμπύλων και γεωμετρικών σχημάτων σε μορφή obj. Η μορφή obj είναι αρκετά γνωστή στη βιομηχανία 3d. Υποστηρίζει τη βασική γεωμετρία καθώς και το υλικό των αντικειμένων. Επομένως τα αντικείμενα που μπορεί να εξάγει μπορεί να είναι:

Πλέγματα: Κορυφές / Προσόψεις / Άκρες / UV 's

Διαχωρισμό βάση Ομάδες / Αντικείμενα

Υλικά / υφές

Καμπύλες και Επιφάνειες

Μόλις επιλέξουμε την εξαγωγή των αντικειμένων θα ζητηθεί ένα όνομα για το αρχείο obj που θα δημιουργηθεί, καθώς και ένα μέρος αποθήκευσης.

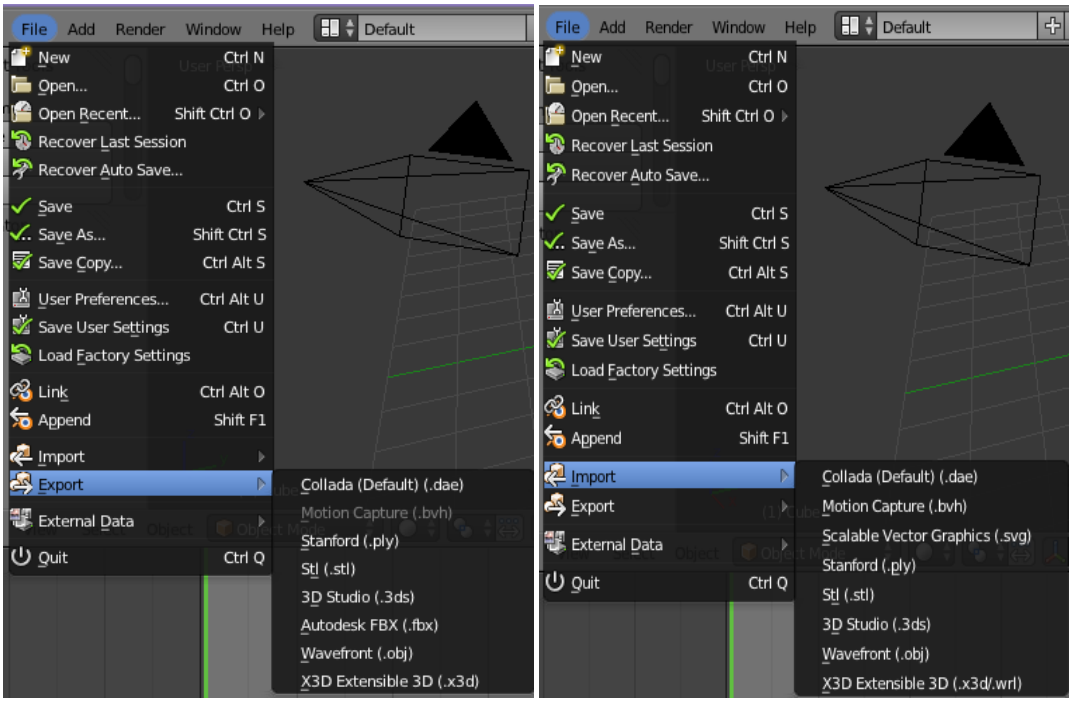

Figure 93 : 4.5.1 Export – Import αντικείμενα

Στην σκηνή στην οποία θα εισάγουμε τα νέα αντικείμενα θα επιλέξουμε και πάλι το μενού File και έπειτα Import→Wavefront (.obj). Έπειτα θα ζητηθεί να αναζητήσουμε το αρχείο obj που αποθηκεύσαμε και αφού το επιλέξουμε το εισάγουμε με το κουμπί Import OBJ. Όλα τα αντικείμενα της προηγούμενης σκηνής , τα οποία τηρούν τις προϋποθέσεις του format obj, θα εισαχθούν στη νέα σκηνή, ώστε να επιλέξουμε αυτά που θα χρησιμοποιηθούν.

4.6 ΔΥΝΑΜΙΚΟ ΚΕΙΜΕΝΟ ΜΕ ΚΩΔΙΚΑ PYTHON

Η γλώσσα Python είναι μια γλώσσα προγραμματισμού που υποστηρίζει διάφορα Προγραμματιστικά Παραδείγματα- programming paradigms. Χαρακτηρίζεται ως διαδραστική, αντικειμενοστρεφής, λειτουργική και επιτακτική γλώσσα προγραμματισμού υψηλού επιπέδου. Παρέχει τη δυνατότητα επαρκούς και αποτελεσματικού προγραμματισμού σε λιγότερες γραμμές κώδικα από ότι μπορεί να απαιτείται σε άλλες γλώσσες προγραμματισμού. Όπως και οι περισσότερες γλώσσες, έτσι και αυτή ενσωματώνει εξαιρέσεις, δυναμικό προγραμματισμό, δυναμικούς τύπους δεδομένων, μεθόδους, κλάσεις, μεταβλητές κτλ. Διαθέτει έναν μεγάλο αριθμό από κατανοητές βιβλιοθήκες για την υλοποίηση των λειτουργιών.

Έχει αναπτυχθεί ως ανοιχτό λογισμικό με τη διαχείρισή της να ανήκει στον μη κερδοσκοπικό οργανισμό Python Software Foundation. Η δημιουργία της ξεκίνησε το 1989 ενώ η έκδοση Python 2.0 καταβλήθηκε το 2000. Η τελευταία έκδοση βγήκε στις 6 Απριλίου 2013 και είναι η 3.3.1.

Πολλές λειτουργίες του Blender μπορούν να υλοποιηθούν με την γλώσσα Python. Οι τομείς Animation, Rendering, Import Export, η Δημιουργία αντικειμένων, αλλά και στο Game Engine, οι λογικές διαδικασίες μπορούν να δημιουργηθούν με script τα οποία δίνουν σε ορισμένες περιπτώσεις παραπάνω δυνατότητες στον χρήστη από ότι τα standard buttons της σουίτας Blender.Για την αλληλεπίδραση με την σουίτα τα script που δημιουργούμε, χρησιμοποιούν το ενσωματωμένο API.

Για να δώσουμε ένα παράδειγμα του πως λειτουργεί ο κώδικας Python στο Blender θα προγραμματίσουμε ένα δυναμικό κείμενο. Ένα κείμενο δηλαδή που θα αλλάζει καθώς το πρόγραμμα θα τρέχει, όπως υλοποιήσαμε με άλλον τρόπο στο κεφάλαιο: Δημιουργία Δυναμικού Κειμένου. Στην περίπτωσή μας θα μετράμε πόσα Monsters έχει σκοτώσει το κανόνι μας.

Στο παιχνίδι μας έχουμε ήδη προσθέσει μια δεύτερη σκηνή την HUD.

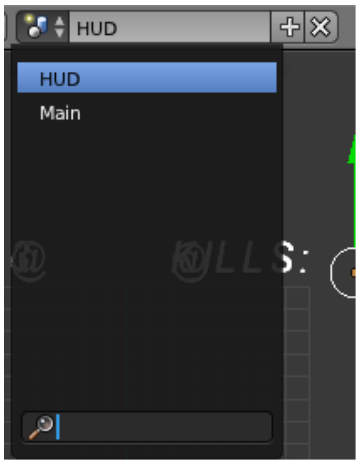

Figure 94 : **4.6.1 Εναλλαγή σκηνής**

Η σκηνή αυτή είναι υπεύθυνη για τα texts που έχουμε επιλέξει να εμφανίζονται στο παιχνίδι. Για το μέτρημα τον σκοτωμένων Monster προσθέτουμε ένα νέο Mesh→ Text με τον συνδυασμό των πλήκτρων Shift+A.

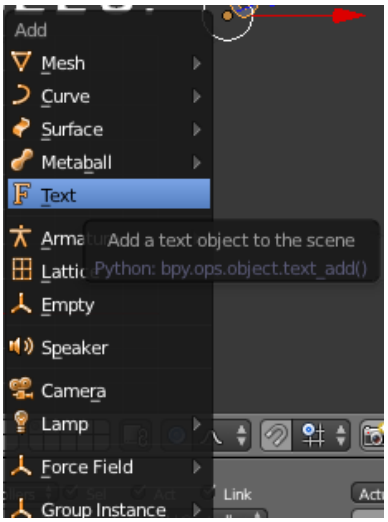

Figure 95 : **4.6.2 Εισαγωγή αντικειμένου Text**

Από την καρτέλα Object Data και στο πεδίο Font, προσθέτουμε την κατάλληλη γραμματοσειρά για το κείμενο μας. Απλά πατάμε το κουμπί OPEN και φορτώνουμε το αρχείο από τον φάκελο fonts.

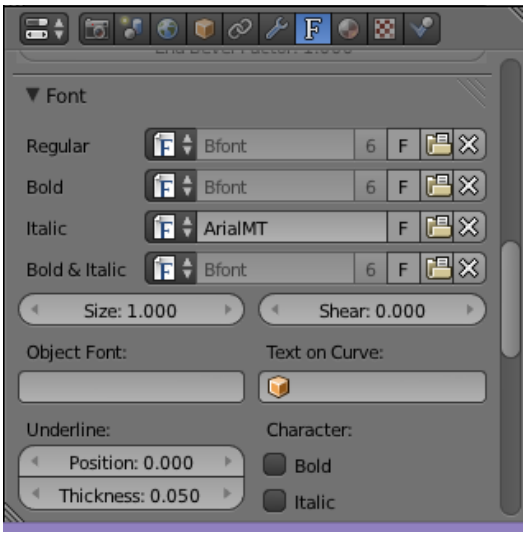

Figure 96 : **4.6.3 Εισαγωγή γραμματοσειράς**

Με το πλήκτρο tab και με backspace σβήνουμε το τρέχον κείμενο και πληκτρολογούμε τον αριθμό 0. Σε αυτό το σημείο θα δώσουμε μια ιδιότητα στο κείμενό μας, αφού γνωρίζουμε πως η δουλειά του θα είναι να μετράει kills. Άρα είναι ανάγκη να το μετατρέψουμε σε Integer. Μεταφερόμαστε επομένως σε Logic Editor και στην καρτέλα Properties πατάμε Add Text Game Property. Από την λίστα των τύπων επιλέγουμε Integer.

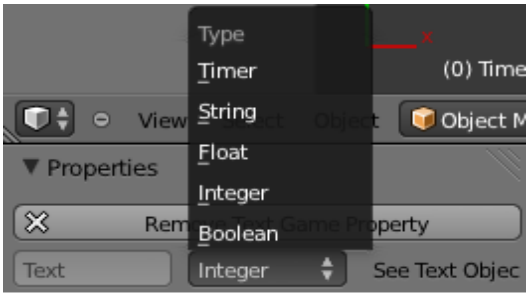

Figure 97 : **4.6.4 Integer Property**

Θα προσθέσουμε μία ακόμη ιδιότητα η οποία θα χρησιμοποιηθεί και επεξηγηθεί στον κώδικα Python. Επομένως από το κουμπί Add Game Property προσθέτουμε την μεταβλητή που ονομάζουμε ‘num_kills’ και της δίνουμε τύπο integer.

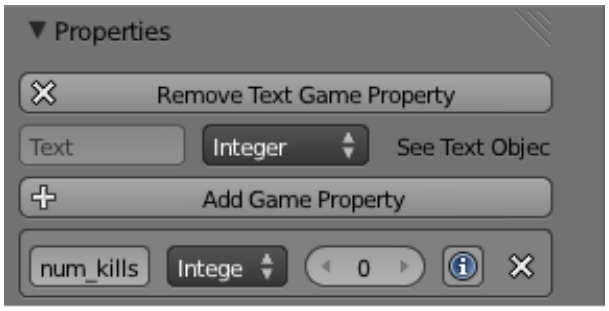

Figure 98 : **4.6.5 Property number of kills**

Στην συνέχεια θα ασχοληθούμε με τον κώδικα Python.

Μεταφερόμαστε επομένως στον Text Editor και επιλέγουμε New, ώστε να προσθέσουμε ένα νέο script. Το μετονομάζουμε σε κάποιο συμβολικό όνομα με την κατάληξη .py και γράφουμε τις εξής γραμμές κώδικα.

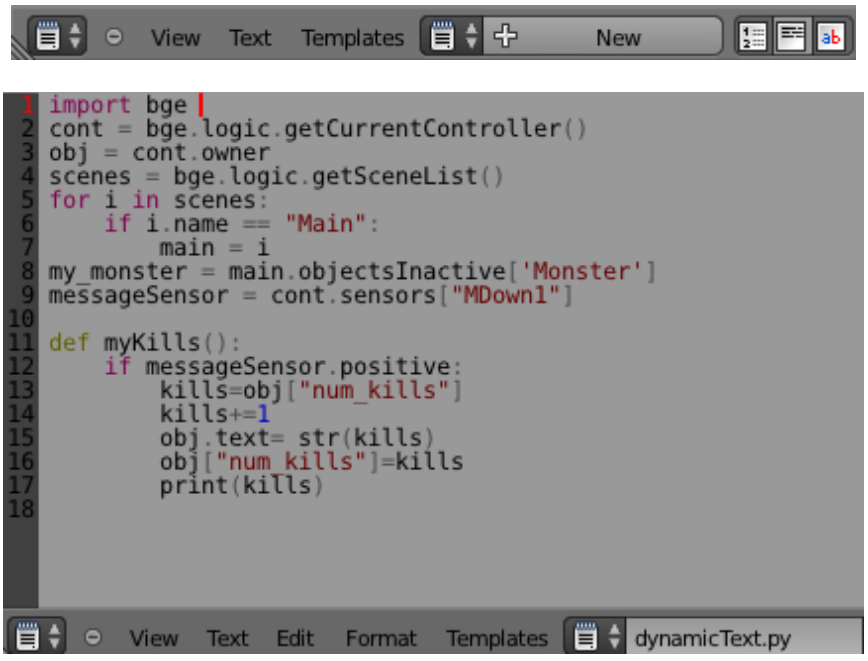

Figure 99 : **4.6.6 Κώδικας Python σε Text Editor**

4.6.1 ΕΠΕΞΗΓΗΣΗ ΚΩΔΙΚΑ PYTHON

```
import bge

# Εισαγωγή του module bge- blender game engine.

cont = bge.logic.getCurrentController()

 # Σε μια μεταβλητή cont αποθηκεύουμε τον controller ο οποίος θα ρυθμίσουμε στην πορεία να καλεί το πρόγραμμα python.

obj = cont.owner

 # Σε μια μεταβλητή obj αποθηκεύουμε τον ιδιοκτήτη του controller που καλέσαμε προηγουμένως. Στην περίπτωσή μας είναι το text που προσθέσαμε στο παιχνίδι.

scenes = bge.logic.getSceneList()   # Σε μια μεταβλητή scenes αποθηκεύουμε τη λίστα με τις σκηνές που έχει το παιχνίδι μας.

for i in scenes:

    if i.name == "Main":

        main = i

# Με την επανάληψη for μέσα στη λίστα των σκηνών, ψάχνουμε με ένα statement If για την σκηνή που λέγεται Main ή όπως αλλιώς έχουμε ονομάσει τη κύρια σκηνή μας. Χρειαζόμαστε τη σκηνή αυτή για να μπορούμε να έχουμε πρόσβαση στο αντικείμενο Monster. Στην τελευταία γραμμή θέτουμε μια μεταβλητή main και εκεί θα αποθηκεύσει το script την σκηνή μόλις την εντοπίσει.

my_monster = main.objectsInactive['Monster']

# Δημιουργία μεταβλητής για την αποθήκευση του αντικειμένου Monster. Από τη σκηνή main παίρνουμε το ανενεργό αντικείμενο που ονομάσαμε Monster (ενδεχομένως και αλλιώς). Επειδή το αντικείμενο αυτό είναι τοποθετημένο σε διαφορετικό layer από το πρώτο, το οποίο είναι και το κυρίαρχο, δεν μπορούμε να το καλέσουμε σαν σκέτο object γιατί δεν θα αναγνωριστεί σαν αντικείμενο της σκηνής.

messageSensor = cont.sensors["MDown1"]

# MDown1 είναι το όνομα του sensor του text,  ο οποίος θα δίνει το ερέθισμα στον controller και θα καλείται το πρόγραμμα. Με αυτή την εντολή καλούμε και κρατάμε τον αισθητήρα αυτόν. Παρακάτω θα επεξηγηθεί η χρησιμότητά του.
```

def myKills():

Η λέξη-κλειδί def σημαίνει definition για τον ορισμό μιας function-λειτουργίας που θα προγραμματιστεί μετά το σύμβολο “ **:** ”. Δώσαμε ένα όνομα και προσθέσαμε κενές παρενθέσεις αφού δεν θα καλέσουμε κάποια παράμετρο μέσω αυτής. Θα παρατηρήσουμε ότι στο σώμα της function όλες οι εντολές ξεκινάνε με Tab. Χωρίς το κενό αυτό θα θεωρηθεί πως βγήκαμε από το σώμα.

    if messageSensor.positive:

Με την εισαγωγή μας στην function ελέγχουμε αν ο αισθητήρας που καλέσαμε πριν είναι positive. Δηλαδή αν δίνει θετικό σήμα εξόδου προς τον controller. Χωρίς αυτόν τον έλεγχο είναι πιθανό ο ελεγκτής python να καλείται παραπάνω από μία φορές. Όλες οι εντολές που ακολουθούνται θα βρίσκονται με Tab μέσα στο σώμα του statement If.

        kills=obj["num_kills"]

Κρατάμε τη μεταβλητή που δημιουργήσαμε στο αντικείμενο text ‘num_kills’, ώστε να χρησιμοποιηθεί ως μετρητής τον οποίο θα αυξάνουμε κάτω από ορισμένες συνθήκες.

        kills+=1

Αυξάνουμε κατά 1 τον αριθμό της μεταβλητής(αφού έχουμε ήδη επιβεβαιώσει ότι δόθηκε θετικό σήμα από τον sensor)

        obj.text= str(kills)

#Αλλάζουμε την έξοδο, δηλαδή το κείμενο που θα φαίνεται του αντικειμένου text στον τρέχοντα αριθμό θανάτων. Επειδή το κείμενο του text δέχεται μόνο String, θα γίνεται κάθε φορά μετατροπή του integer.

        obj["num_kills"]=kills

#Στην ιδιότητα-μετρητή βάζουμε τη νέα τιμή μετά την αύξηση.

        print(kills)

#Εντολή εκτύπωσης για την βοήθεια του χρήστη, ως προς τον έλεγχο του αποτελέσματος στην κονσόλα(Python Console). Μπορεί και να παραλειφθεί.

4.6.2 LOGIC EDITOR ΤΟΥ ΑΝΤΙΚΕΙΜΕΝΟΥ TEXT

Έχοντας τελειώσει με τον κώδικα, ξαναγυρνάμε σε Logic Editor type και προσθέτουμε τα logic bricks που θα χρησιμεύσουν για να καλέσουμε το script που μόλις δημιουργήσαμε.

Με επιλεγμένο το αντικείμενο text προσθέτουμε τον sensor→ Message. Με τον συγκεκριμένο αισθητήρα θα αναζητάμε ένα συγκεκριμένο μήνυμα και όταν εντοπίζεται θα ενεργοποιείται ο ελεγκτής. Το μήνυμα το οποίο θα προσθέσουμε στο πεδίο Subject είναι: MonsterDown. Αφορά το μήνυμα που δημιουργήσαμε στο αντικείμενο Monster, για κάθε φορά που το πετυχαίνει κάποια σφαίρα(Sensor-Collision) στη σκηνή Main.

Σαν Ελεγκτή θα προσθέσουμε τον Python. Θα επιλέξουμε από την λίστα των script types, τον τύπο Module και στο κενό θα πληκτρολογήσουμε το όνομα του script και θα καλέσουμε την function που ορίσαμε μέσα σε αυτό. Στην περίπτωσή μας δηλαδή θα πληκτρολογήσουμε: dynamicText.myKills.

Figure 100 : 4.6.2.1 Καλώντας το module του κώδικα Python

ΣΥΜΠΕΡΑΣΜΑΤΑ

Το Game Engine της ανοιχτής σουίτας Blender δίνει πάρα πολλές επιλογές στον σχεδιαστή του παιχνιδιού. Αν και δύσκολο στην κατανόηση συνδυάζει κουμπιά και μενού με κώδικα Python για ένα ακριβέστερο και καλύτερο αποτέλεσμα. Διαθέτει δικιά του μηχανή μόνο για αυτόν τον σκοπό, με εργαλεία χρήσιμα για την δημιουργία παιχνιδιών.

Παρόλα αυτά είναι δύσκολο στην εκμάθησή του και απαιτεί συνεχή εξάσκηση για την τελειοποίηση και την ολοκλήρωση των γνώσεων του σχεδιαστή. Θεωρείται κατά κάποιον τρόπο παρεξηγημένο, αφού οι αρχάριοι χρήστες θα επιλέξουν ένα πιο εύκολο πρόγραμμα, χωρίς να κατανοούν τις δυνατότητες που τους προσφέρει το τρέχον. Εξάλλου ο προγραμματισμός σε Python είναι πολύ πιο απλός και κατανοητός από τις διάφορες γλώσσες που χρησιμοποιούν οι άλλες μηχανές.

Κατά τη διάρκεια κατασκευής ενός παιχνιδιού θα εμφανιστούν προβλήματα, όπως η προσωρινή εξαφάνιση αντικειμένων, ή η δυσκολία ‘zoom in’ σε μια σκηνή, που χρειάζονται υπομονή, εμπειρία και έρευνα για την επίλυσή τους. Πολύ σημαντικό είναι ο χρήστης του BGE να κατανοήσει τους άξονες x,y,z για την σωστή μετακίνηση των αντικειμένων που επιθυμεί.

Ο χρήστης είναι υποχρεωμένος να αποθηκεύει συνεχώς τη δουλειά του, αφού το Blender θεωρεί πως είναι ικανός να κατανοεί πότε χρειάζεται να αποθηκεύσει, με αποτέλεσμα να μην υπάρχει αντίστοιχη ερώτηση με την έξοδο από τη σουίτα. Η παράλειψή του θα φέρει δυσκολία στην επανάκτηση της προόδου που μπορεί να σημειώθηκε.

Στο διαδίκτυο οι σελίδες που αφορούν το BGE έχουν αυξηθεί σημαντικά τα τελευταία χρόνια. Καθώς εξελίσσεται, οι χρήστες αναζητάνε τις γνώσεις που μπορεί να προσφέρει. Υπάρχουν όμως παράπονα για την σκόρπια τοποθέτηση των κειμένων. Δυστυχώς αν και υπάρχουν λίγα εκπαιδευτικά υλικά, τα περισσότερα είναι σκόρπια ή παλαιότερα και οι νέες εκδόσεις δεν επεξηγούνται σωστά.

Όσον αφορά τον σχεδιασμό σε Blender είναι καλό να θυμόμαστε πως ένα παιχνίδι δεν αποτελείται μόνο από την κεντρική σκηνή. Η σουίτα δίνει τη δυνατότητα δημιουργίας πολλαπλών σκηνών και ένωσης μεταξύ αυτών, για τη μετάβαση από τη μία στην άλλη. Animation και γραφικά ενώνονται με το BGE με σκοπό μιας αληθοφανούς κατασκευής.

Φυσικά η δημιουργία ενός παιχνιδιού δεν τελειώνει εδώ, αλλά αφήνεται στην φαντασία του κάθε σχεδιαστή, το κλειδί για το ξεδίπλωμα των δυνατοτήτων της σουίτας με στόχο ένα πολυσύνθετο και ενδιαφέρον αποτέλεσμα.

ΕΠΙΛΟΓΟΣ

Η σουίτα γραφικών Blender έχει πλέον εδραιωθεί στον χώρο της δημιουργία γραφικών, από το απλό animation ως τη σύνθετη κατασκευή παιχνιδιού. Οι χρήστες του αυξάνονται χρόνο με το χρόνο και πολλοί διακρίνονται στις ικανότητές τους στα γραφικά μέσω του προγράμματος αυτού. Απόδειξη αυτού το γνωστό animation “Big Buck Bunny” αλλά και τα παιχνίδια “Yo Frankie” και “Super BlenderGalaxy”. Χρειάστηκαν ομάδες προγραμματιστών σε Blender για την κατασκευή τους και το αποτέλεσμα άξιζε τον κόπο.

Figure 101 : **Big Buck Bunny – Yo Frankie**

Ένα πολύ σημαντικό έργο είναι η τέταρτη ταινία μικρού μήκους από σχεδιαστές 3D γραφικών σε Blender, που λέγεται “Tears of Steel” , ένα πολύ καλό παράδειγμα των δυνατοτήτων σχεδίασης σε Blender και ενσωμάτωσης τους σε ταινία.

Figure 102 : **Tears of Steel**

Οι εκδόσεις του Blender γίνονται όλο και καλύτερες, αφού δεν βελτιώνουν απλά τα υπάρχοντα εργαλεία, αλλά προσθέτουν και καινούργιες προοδευτικές λειτουργίες που προβλέπεται πως θα τραβήξουν όλο και περισσότερο κόσμο στον σχεδιασμό γραφικών.

Η τελευταία έκδοση του Blender(2.67) δίνει πλέον την δυνατότητα δημιουργίας 2D γραφικών σε περιβάλλον 3D για την κατασκευή cartoon. Έχει αποδώσει μια νέα μηχανή με όνομα Freestyle για τη δημιουργία απλών γραμμών. Έχει βελτιώσει την κίνηση της κάμερας βάση του αντικειμένου, σε πιο γρήγορα και ακριβή. Τα εργαλεία χρωματισμού αντικειμένων εμπλουτίστηκαν καθώς προστέθηκαν νέες Rendering δυνατότητες. Μα το πιο ενδιαφέρον και δυνατό σημείο της καινούργιας αυτής έκδοσης είναι η προσθήκη εργαλείων για τον σχεδιασμό αντικειμένων που μπορούν να εκτυπωθούν σε 3D εκτυπωτές.

Figure 103 : **2.67 Blender**

Σαν καινούργιος χρήστης του Blender, και κυρίως της μηχανής παιχνιδιών θεωρώ πως είναι ένα πολύ δυνατό εργαλείο που θα εξελιχθεί σε ακόμη καλύτερο και θα κερδίσει τον κόσμο. Υπάρχουν όμως ακόμη σημεία που χρειάζονται προσοχή και διόρθωση και σίγουρα ένα από αυτά είναι η σύνθετη και πολύπλοκη δομή του. Τα διάφορα κουμπιά και μενού μπορούν πολύ εύκολα να μπερδέψουν τον σχεδιαστή και μια προσεκτική ομαδοποίηση τους θα ήταν πολύ χρήσιμη, ειδικά για αρχάριους που θέλουν να εκπαιδευτούν.

Ένα είναι σίγουρο πως η σουίτα Blender έχει έρθει για να μείνει και να συναγωνιστεί πολλές άλλες(ανοιχτές και μη), ίσως και να τις ξεπεράσει, αφού κάθε έκδοσή της καλύπτει τομείς, με τους οποίους ο σχεδιαστής θα βρει ευκολία στη δουλειά του και κίνητρο για περαιτέρω δημιουργική ανάπτυξη και εκμετάλλευση των δυνατοτήτων του.

ΒΙΒΛΙΟΓΡΑΦΙΑ

- Δημιουργία και επεξεργασία τρισδιάστατων γραφικών μέσω της ανοιχτής σουίτας Blender **των Φοιτητών Βασιλακοπούλου Στεργιανή και Παπαδόπουλος Δημήτριος(2012)**

Ιστότοποι

Επίσημος οδηγός του Blender :

http://wiki.blender.org/index.php/Doc:2.6/Manual

Tutorials for Blender:

http://www.tutorialsforblender3d.com/

Blender Cookie:

http://cgcookie.com/blender/

Materials and Textures in Blender (pdf):
http://www.cogfilms.com/tutorials/Cog_s_BSoD/Introduction_to_Materials_and_Procedural_Textures.pdf

Sensor Pulse:

http://gameblender.wikia.com/wiki/Sensors

Channel Tutorials Goran Milovanovic:

http://www.youtube.com/user/goranmilovano

Channel Tutorials Ira Krakow:

http://www.youtube.com/user/irakrakow

WikiBook:

http://en.wikibooks.org/wiki/Blender_3D:_Noob_to_Pro

Explosion :
http://www.youtube.com/watch?v=nIhjdpkAfmA&NR=1&feature=endscreen

Python API Documentation:
http://www.blender.org/documentation/blender_python_api_2_64_release/contents.html

Python Scripting for the Game Engine:

http://www.cgmasters.net/free-tutorials/python-scripting/

Real-time Font:

http://bgetutorials.wordpress.com/2007/11/07/how-to-get-realtime-text-in-blender/

Usage of ftBlender:

http://www.blendenzo.com/tutFTBlender.html

Game Engines:
http://www.worldofleveldesign.com/categories/level_design_tutorials/recommended-game-engines.php