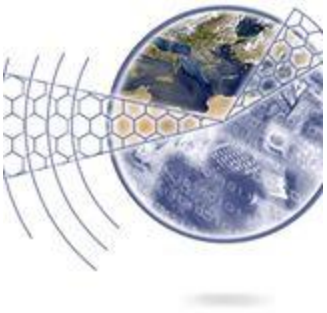
Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Μηχανικών Πληροφορικής



Πτυχιακή εργασία

Δημιουργία τρισδιάστατων γραφικών με τη χρήση του SketchUp

Σαββιδάκη Δέσποινα ΑΜ: 2041

Επιβλέπων καθηγητής: Μαλάμος Αθανάσιος

ΗΡΑΚΛΕΙΟ 2013

Ευχαριστίες

Ευχαριστώ πάρα πολύ τον Αντρέα και τον Κώστα για την πολύτιμη βοήθεια που μου πρόσφεραν κατά τη διάρκεια υλοποίησης της εργασίας μου.

Σύνοψη

Σε αυτή την πτυχιακή εργασία θα ασχοληθούμε με τη δημιουργία τρισδιάστατων γραφικών, και την εξέλιξή τους σε μία απλή διαδραστική εφαρμογή, δηλαδή ένα παιχνίδι.

Αναφέρουμε περιληπτικά παρόμοια παιχνίδια που έχουν ήδη δημιουργηθεί, καθώς και διάφορα προγράμματα τα οποία είναι κατάλληλα για σχεδιασμό παιχνιδιών.

Για τον σχεδιασμό των βασικών αντικειμένων θα χρησιμοποιήσουμε την εφαρμογή SketchUp, ενώ για την δημιουργία του παιχνιδιού την εφαρμογή Blender.

Θα δούμε αναλυτικά τα εργαλεία και των δύο εφαρμογών, τις δυνατότητες τους καθώς και το πώς μπορούν να συνεργαστούν.

Τέλος θα χρησιμοποιήσουμε ένα script σε γλώσσα Python, μέσω της εφαρμογής Blender για τη δημιουργία κίνησης.

Abstract

In this paper we will deal with the creation of three-dimensional graphics, and their development in a simple interactive application, i.e. a game.

We summarize similar games that have already been created, and various programs which are suitable for game design.

For the design of the basic objects will use the SketchUp software. For the development to a game, we will use the Blender software.

We will examine the tools that both softwares have, their capabilities and how they can cooperate.

Finally we will use a Python script, through the Blender software to give motion.

Περιεχόμενα

Πίνακας εικόνων

1. ΕΙΣΑΓΩΓΗ

1.1 Γενικά

Το αντικείμενο με το οποίο ασχοληθήκαμε στην παρούσα πτυχιακή εργασία, είναι ο σχεδιασμός τρισδιάστατων γραφικών και στη συνέχεια η δημιουργία ενός παιχνιδιού με βάση το μοντέλο που φτιάξαμε.

Για να υλοποιήσουμε τα παραπάνω χρησιμοποιήσαμε τις εφαρμογές SketchUp για την σχεδίαση του βασικού μοντέλου, πάνω στο οποίο θα στηριχτεί το παιχνίδι. Για τη δημιουργία του παιχνιδιού χρησιμοποιήσαμε την εφαρμογή Blender.

1.2 Σκοπός

Αρχικά ο σκοπός, ήταν η δημιουργία ενός τρισδιάστατου μοντέλου, δείχνοντας έτσι τις δυνατότητες του SketchUp. Επειδή όμως αυτές είναι περιορισμένες, σε αρχιτεκτονικό και δομικό επίπεδο, αποφασίστηκε να εξελιχθεί το μοντέλο, σε παιχνίδι, με τη βοήθεια του προγράμματος Blender.

1.3 Περιγραφή

Το μοντέλο που δημιουργήσαμε, αναπαριστά ένα δωμάτιο το οποίο έχει ένα τραπέζι με ένα ενυδρείο επάνω.

Το παιχνίδι ξεκινά κάνοντας μία πλοήγηση στο δωμάτιο και στη συνέχεια η δράση μεταφέρεται μέσα στο ενυδρείο, όπου ο παίχτης έχει τη δυνατότητα να πιάσει τα ψάρια που υπάρχουν μέσα σε αυτό είτε με ένα καμάκι που εκτοξεύεται κάθε φορά, είτε με ένα αγκίστρι. Υπάρχει ένας μετρητής πόντων, και για κάθε ψάρι που πιάνεται οι πόντοι θα αυξάνονται κατά έναν.

Η πλοήγηση μέσα και έξω από το ενυδρείο γίνεται με τη χρήση των πλήκτρων W, S για μπρος και πίσω αντίστοιχα, και με το ποντίκι στρίβουμε δεξιά, αριστερά, πάνω και κάτω. Πατώντας το αριστερό κουμπί του ποντικιού εκτοξεύεται το καμάκι, ενώ με τα πλήκτρα πάνω, κάτω, δεξιά, αριστερά, A και Z κινούμε το αγκίστρι.

1.4 Δομή

- **Κεφάλαιο 1: Εισαγωγή**
- **Κεφάλαιο 2: Video Games και Game Engines**
  Αναφέρουμε τα video games ανάλογα με τον τύπο τους και τις Game Engines που τα δημιουργουν.
- **Κεφάλαιο 3: Διαδικασίες Ανάπτυξης**
  Αναφορά στα εργαλεία που χρησιμοποιήθηκαν και ανάλυση των διαδικασιών μοντελοποίησης.
- **Κεφάλαιο 4: Αποτελέσματα**
  Παρουσιάζουμε τους τρόπους με τους οποίους μπορούμε να κάνουμε απόδοση του παιχνιδιού και αναφέρουμε τα συμπεράσματα μας σχετικά με την αποτελεσματικότητα των προγραμμάτων.
- **Παράρτημα**
  SketchUp
  Δημιουργία του αρχικού μοντέλου στο SketchUp και δοκιμάζουμε τα materials και styles του.
  Import – Export
  Πειραματιζόμαστε με την εξαγωγή του μοντέλου από το SketchUp και την εισαγωγή του στο Blender, έτσι ώστε να διαλέξουμε τις καταλληλότερες ρυθμίσεις, για να έχουμε την καλύτερη δυνατή μεταφορά του μοντέλου από τη μία εφαρμογή στην άλλη.
  Blender
  Σχεδιασμός περαιτέρω αντικειμένων και χρήση διάφορων λειτουργιών που διαθέτει για πιο ρεαλιστικά αποτελέσματα. Δημιουργία διαδραστικότητας με την μηχανή Game Engine. Χρήση γλώσσας προγραμματισμού Python.

2. VIDEO GAMES ΚΑΙ GAME ENGINES

2.1 Εισαγωγή

Ένα video game είναι ένα ηλεκτρονικό παιχνίδι, το οποίο συνδυάζει την αλληλεπίδραση μεταξύ του ανθρώπου και ενός περιβάλλοντος χρήστη, ώστε να δημιουργηθεί μία οπτική ανταπόκριση σε μία τηλεοπτική συσκευή. Τα ηλεκτρονικά συστήματα που χρησιμοποιούνται για τα video games ονομάζονται πλατφόρμες, τέτοια συστήματα είναι οι προσωπικοί υπολογιστές και οι κονσόλες παιχνιδιών. [1]

Με τον όρο πλατφόμα αναφερόμαστε στον συνδυασμό ηλεκτρονικών εξαρτημάτων ή hardware του υπολογιστή, όπου μαζί με το λογισμικό επιτρέπουν ένα video game να λειτουργήσει. [2] Πλέον χρησιμοποιείται επίσης για να δηλώσει την ηλεκτρονική υπηρεσία εντός της οποίας παίζεται το παιχνίδι, παράδειγμα η σελίδα κοινωνικής δικτύωσης "Facebook".

Οι συσκευές εισόδου που χρησιμοποιούνται για τον χειρισμό των video games διαφέρουν ανάλογα με την πλατφόρμα που χρησιμοποιείται, παραδείγματα είναι τα χειριστήρια, joysticks, gamepads, πληκτρολόγιο, ποντίκι.

2.2 Είδη video games

2.2.1 Εισαγωγή

Τα video games χωρίζονται σε είδη ανάλογα με τις μεθόδους, τους στόχους, την τεχνοτροπία, την διαδραστικότητα τους με τον παίχτη. Με την εξέλιξή τους όμως παρατηρούμε ότι τα σημερινά video games είναι μία μίξη των διάφορων ειδών που υπάρχουν. [3]

2.2.2 Είδη [4]

Τα βασικά είδη των video games και οι κατηγορίες που χωρίζονται φαίνονται στον παρακάτω πίνακα.

Τα πιο διαδεδομένα είναι τα first person shooter και third person shooter παιχνίδια όπως το Doom το οποίο βγήκε πρώτη φορα σε κυκλοφορία το 1993, μέχρι και το 2007 όπου κυκλοφόρησε το Doom4 και ήταν first person shooter παιχνίδι επιστημονικής φαντασίας και τρόμου.
Το Call Of Duty που ήταν και first person και third person shooter που κυκλοφόρησε το 2003 και είχε σαν μοτίβο τον Β΄ παγκόσμιο πόλεμο, η εταιρεία Activision κυκλοφορήσε και άλλες σειρές του παιχνιδιού μέχρι το 2014.
Το Resident Evil επίσης παιχνίδι τρόμου, επιβίωσης και φαντασίας, το 1996 ενώ και μέχρι το 2012 η εταιρία Capcorm κυκλοφορούσε μία καινούρια έκδοσή του, κάθε χρόνο.

Επίσης διαδεδομένα είναι τα παιχνίδια ρόλων πολλαπλών χρηστών MMORPG, όπως το World of Warcraft (WoW) της Blizzard Entertainment ένα παγκόσμιο παιχνίδι, όπου οι παίκτες ελέγχουν ένα avatar χαρακτήρα και έχουν σκοπό να πολεμήσουν τέρατα και να εξερευνήσουν τοπία, ολοκληρώνοντας διάφορες αποστολές.

Παιχνίδια MMORPG αλλά και Real Time Strategy είναι το Warcraft της Blizzard Entertainment, όπου οι αντίπαλοι παίκτες ελέγχουν τον χαρακτήρα τους και δίνουν εντολές μάχης σε εικονικούς στρατούς.

Το πιο γνωστό παιχνίδι προσωμοιώσης του πραγματικού κόσμου είναι το The Sims, της Electronic Arts. Κυκλοφόρησε τον Φλεβάρη του 2000 η πρώτη σειρά και η τελευταία τον Οκτώβρη του 2013.

Το Pro Evolution Soccer της εταιρείας Konami και το FIFA Soccer της Electronic Arts είναι τα πιο διάσημα Sports game, όπου προσωμοιώνουν ποδοσφαιρικούς αγώνες. Το Pro Evolution Soccer κυκλοφόρησε πρώτη φορά το 2001, ενώ το 2013 κυκλοφόρησε η τελευταία έκδοσή του. Το FIFA Soccer κυκλοφόρησε το 1993 και το 2013 κυκλοφόρησε το τελευταίο FIFA 14.

<table>
<tr><th>Είδη</th><th colspan="5">Κατηγορίες</th></tr>
<tr><td rowspan="5">Action</td><td rowspan="5">Fighting
Μάχη έναν προς ένα, δύο χαρακτήρων
Πχ. Super Smash Bros, Street Fighter II, Mortal Kombat</td><td rowspan="5">Maze
Το πεδίο είναι ένας λαβύρινθος όπου πρέπει να διασχίσει ο παίχτης ή οι παίχτες, έχοντας εμπόδια και χρόνο
Πχ. Pac-Man</td><td colspan="3">Shooter
Μάχες όπου συμμετέχουν όπλα και πυραύλοι</td></tr>
<tr><td colspan="3">First-person shooter [5]
απο την οπτική γωνία του χαρακτήρα που ελέγχει ο παίχτης
Πχ. Doom, Call of Duty, Battlefield</td></tr>
<tr><td colspan="3">Third-person shooter
απο μια οπτική γωνία όπου ο παίχτης βλέπει απο πίσω τον χαρακτήρα
Πχ. Resident Evil 4, Mass Effect, Grand Theft Auto</td></tr>
<tr><td colspan="3">Shoot 'em up
παίκτης ελέγχει έναν χαρακτήρα ή όχημα και πυροβολεί μεγάλο αριθμό των εχθρών, arcade παιχνίδια</td></tr>
<tr><td colspan="3">MMO FPS [6]
Ανάμιξη των fps με τα παιχνίδια πολλαπλών χρηστών. Συνήθως είναι παιχνίδια που παίζονται στο διαδίκτυο, με παίκτες απο όλον τον κόσμο, που συναντιούνται σε έναν εικονικό</td></tr>
<tr><td>Adventure</td><td>Real-time 3D adventures
3D παιχνίδια που ακολουθούν τους φυσικούς νόμους</td><td>Text adventures
διαδραστική μυθοπλασία με εντολες απο το πληκτρολογιο</td><td>Graphic adventures
εξελιξη του προηγούμενου, με διαδράστικότητα μεσω ποντικιου</td><td colspan="2">Visual novels
Διαδραστικά παιχνίδια φαντασίας με στατικά γραφικά και συνήθως anime στυλ, τα οποία μοιάζουν με μυθιστορήματα</td></tr>
<tr><td>Role-Playing
Ο παίκτης αποκτά ένα ρόλο και κινείται με βάση ένα προκαθορισμένο σενάριο</td><td>Western RPGs and Japanese RPGs
Με βάση την αμερικάνικη και ιαπωνική κουλτούρα</td><td>Use of fantasy in RPGs
Σε φανταστικό ή φουτουριστικό περιβάλλον</td><td colspan="3">MMORPGs [7]
Μίξη των παιχνιδιών πολλαπλών παιχτών με RP Games. Παίκτες απο όλο τον κόσμο αναλαμβάνουν κάποιον χαρακτήρα και αλληλεπιδρούν μεταξύ τους μέσα σε ένα εικονικό κόσμο.
Πχ. World of Warcraft (WoW), Star Wars: The Old Republic, Final Fantasy XI, The Lord of the Rings Online</td></tr>
<tr><td colspan="6">Simulation
αναπαράσταση πραγματικότητας ή πλασματικής πραγματικότητας, πόλεων και ανθρώπων που ζούν σε αυτές, την καθημερινότητα τους, ή αναπαράσταση οδήγησης μεταφορικών μέσων όπως αεροπλάνων και τρένων
πχ.The Sims</td></tr>
<tr><td>Strategy</td><td>4X game
στρατηγικής με 4 βασικούς στόχους eXplore, eXpand, eXploit and eXterminate</td><td>Artillery game
Παιχνίδια 2 -3 παιχτών όπου δίνουν μάχες</td><td colspan="3">Real-time strategy
Οι παίκτες πρέπει να πάρουν αποφάσεις και να δράσουν, όσο το παιχνίδι εξελίσεται
Πχ. Warcraft series, Age of Empires series, Dawn of War, Command and Conquer</td></tr>
<tr><td>Sports</td><td>Racing
Αναπαραστάσεις αγώνων ταχύτητας</td><td colspan="4">Sports game
Αναπαράσταση διάφορων σπορ
Πχ. FIFA Soccer, Pro Evolution Soccer</td></tr>
<tr><td>Other</td><td>Music
Πχ. Guitar Hero</td><td>Puzzle
Πχ. Tetris</td><td>Trivia
Παιχνίδια γνώσεων</td><td colspan="2">Casual
Απλά παιχνίδια που απευθύνονται σε ένα μαζικό κοινό
Πχ. Mahjong, ναρκαλιευτής, πασιέντζα</td></tr>
<tr><td>By purpose</td><td>Exergame
Πχ. Wii Fit</td><td>Educational</td><td>Art</td><td>Casual</td><td>Serious
Παιχνίδια σχεδιασμένα για εκπαίδευση, άμυνα, επιστημονική έρευνα, υγειονομική περίθαλψη, διαχείριση κινδύνου, θρησκεία Προσομοιώσεις του πραγματικού κόσμου, ή παιχνίδια μεθόδων</td></tr>
</table>

Είδη και κατηγορίες των video games [8]

2.3 3D προγράμματα

2.3.1 Εισαγωγή

Μια game engine είναι ένα σύστημα σχεδιασμένο να δημιουργεί και να αναπτύσσει video games. Οι κορυφαίες μηχανές παιχνιδιών παρέχουν ένα πλαίσιο λογισμικού που χρησιμοποιείται απο προγραμματιστές για να δημιουργήσουν παιχνίδια για προσωπικούς υπολογιστές, κονσόλες παιχνιδιών, κινητά τηλέφωνα. Η βασική λειτουργικότητα που τυπικά προέρχεται από μια μηχανή παιχνιδιού περιλαμβάνει ένα μηχανισμό απόδοσης για δισδιάστατα ή τρισδιάστατα γραφικά, μία

μηχανή προσωμοίωσης φυσικών καταστάσεων ή μηχανή ανίχνευσης σύγκρουσης, τεχνητή νοημοσύνη, ήχο, scripting, animation, δικτύωση, streaming, threading και γράφημα σκηνής.[8]

Οι μηχανές τρισδιάστατων γραφικών ή τα συστήματα απόδοσης τους, συνήθως βασίζονται σε ένα περιβάλλον προγραμματισμού γραφικών (API), όπως το Direct3d ή το OpenGL.[9]

2.3.2 Game Engines

- **Unreal Engine** [10]

Αναπτύχθηκε απο την από την Epic Games. Εμφανίζεται πρώτη φορά με το ομόνυμο first person shooter παιχνίδι το 1998. Παρότι είχε σχεδιαστεί για για FPS παιχνίδια, έχει χρησιμοποιηθεί με επιτυχία και σε MMORPG και RPG παιχνίδια. Γραμμένη σε C++ διαθέτει υψηλό βαθμό φορητότητας και σήμερα χρησιμοποιείται απο πολλούς προγραμματιστές.
Η τρέχουσα έκδοσή του είναι η Unreal Engine 3 η οποία εχει σχεδιαστεί για Windows και Xbox 360, Windows Vista, Windows 7, Linux, PlayStation 3, Wii U, και iOS, για Android, για το Adobe Flash Player 11 και για HTML5.
Έχει χρησιμοποιηθεί για το παιδικό TV show LazyTown, καθώς και για τα παιχνίδια Unreal (FPS), Mass Effect (Action role-playing, third-person shooter), Medal of Honor: Airborne (FPS), Star Trek: Deep Space Nine: The Fallen (Third Person Shooter, Adventure), Sephiroth (MMORPG).

- **Doom** [11]

Ο πηγαίος κώδικας για την έκδοση Linux του Doom κυκλοφόρησε στο κοινό στις 23 Δεκεμβρίου 1997. Αργότερα επανακυκλοφόρησε υπό την GNU General Public License, το 1999.
Παρόλο που η μηχανή αποδίδει ένα τρισδιάστατο χώρο, αυτός ο χώρος προβάλεται απο μία δισδιάστατη κάτοψη. Η μηχανή Doom αργότερα μετονομάστηκε σε id Tech 1 για να φτάσει σήμερα στο id Tech 4.
Το id Tech 4 είναι γραμμένο σε γλώσσα C++. Υποστηρίζει bump mapping,normal mapping, και specular highlighting, δυναμικά εφέ, skeletal animation. Διαθέτει MegaTexture τεχνολογία απόδοσης, γλώσσα προγραμματισμού για scripting παρόμοια με την C++, υποστηρίζει ήχο και χρησιμοποιεί το μοντέλο client–server.
Παιχνίδια που έχουν δημιουργηθεί είναι τα Doom, Doom II, Final Doom, Doom3.

- **Anvil** [12]

Δημιουργήθηκε το 2007 απο την εταίρεια προγραμματισμού video game, Ubisoft Montreal για να χρησιμοποιείται απο τα Microsoft Windows, PlayStation 3, Xbox 360, Wii U, PlayStation Vita, Xbox One, και PlayStation 4. Σύμφωνα με τον Claude Langlais, η μοντελοποίηση του περιβάλλοντος γίνεται στο 3ds Max και και στο ZBrush για τους χαρακτήρες.
Παιχνίδια που έχουν δημιουργηθεί: Assassin's Creed II, Prince of Persia: The Forgotten Sands, Assassin's Creed: Brotherhood, Assassin's Creed: Revelations.

- **Crystal_Tools** [13]

Συνδυάζει πρότυπες βιβλιοθήκες για στοιχεία όπως γραφικά, ήχο και τεχνητή νοημοσύνη, και παρέχει στους προγραμματιστές παιχνιδιών διάφορα εργαλεία συγγραφής. Οι στόχοι της είναι το PlayStation 3, το Xbox 360, η Microsoft Windows και το Wii.
Συνδυάζει τυπικές βιβλιοθήκες για την απόδοση των γραφικών , επεξεργασίας φυσικής , έλεγχος κίνησης , κινηματογραφικά , οπτικά εφέ , ήχου , τεχνητή νοημοσύνη και τη δικτύωση. Περιλαμβάνει ένα πρόγραμμα προβολής χαρακτήρων για 3D μοντέλα, για κόψιμο σκηνών, εργαλείο προ-οπτικοποιήσης, και δημιουργίας ήχου. Η Χρήση τρίτων προγραμμάτων όπως Autodesk Maya, Autodesk Softimage και το Adobe Photoshop υποστηρίζεται μέσω plug-ins.
Παιχνίδια: Final Fantasy XIII, Final Fantasy Versus XIII, Final Fantasy XIV, Final Fantasy XII-2.

- **Panda3D** [14]

Είναι μια μηχανή δημιουργίας γραφικών σκηνής, περιλαμβάνει γραφικά, ήχο, σύστημα εισόδου/εξόδου, ανίχνευση σύγκρουσης και άλλες ιδιότητες που χρειάζονται για τη δημιουργία 3D. Είναι ένα ελεύθερο λογισμικό ανοιχτού κώδικα. Η προτιθέμενη γλώσσα για την ανάπτυξη του παιχνιδιού ειναι η python. Η μηχανή ειναι γραμμενη σε C++.

Έχει μη γραφικές δυνατότητες οι οποίες είναι: Εργαλεία ανάλυσης απόδοσης, Εργαλεία εξερεύνησης γραφήματος, Εργαλεία διόρθωσης σφαλμάτων, τρισδιάστατος ήχος, Μηχανή δυναμικών, Υποστήριξη συσκευών εισοδου/εξοδου, Τεχνητη νοημοσύνη.
Έχει χρησιμοποιηθεί για τρισδιάστατα θεματικά πάρκα της Disney και στη δημιουργία παιχνιδιών, για το δευτερο MMORPG, Pirates of the Caribbean Online.

- **Maya** [15]

Είναι λογισμικό για τρισδιάστατα γραφικά υπολογιστών που τρέχει σε Windows, Mac OS και Linux, αναπτύχθηκε αρχικά από Alias Systems Corporation και επί του παρόντος ανήκει και αναπτύχθηκε από την Autodesk, Inc. Χρησιμοποιείται για τη δημιουργία διαδραστικών εφαρμογών 3D, συμπεριλαμβανομένων των βιντεοπαιχνιδιών, ταινίες κινουμένων σχεδίων, τηλεοπτικές σειρές και οπτικά εφέ.
Συστατικά: εφέ υγρών, ρούχων, γουνας, τριχών
Έχει χρησιμοποιηθεί για τη δημιουργία γραφικών για πολλές κινηματογραφικές ταινίες, συμπεριλαμβανομένων των «Το Κορίτσι με το Τατουάζ», «Avatar (2009)», «Ψάχνοντας τον Νέμο», «Up», «Monsters», επίσης, χρησιμοποιείται για τη δημιουργία οπτικών εφέ για τηλεοπτικά προγράμματα, όπως τα «Game of Thrones», «The Walking Dead», «Once Upon a Time», «Bones», «Futurama», «Boardwalk Empire» και «South Park».
Συμμετείχε στη δημιουργία των οπτικών εφέ για βιντεοπαιχνίδια, συμπεριλαμβανομένων Halo 4.

- **Unity** [16]

Ειναι μια cross-platform game engine με ενσωματωμένο IDE. αναπτύχθηκε από Unity Technologies. χρησιμοποιείται για την ανάπτυξη video games για web plugins, πλατφόρμες desktop, κονσόλες και φορητές συσκευές.
Προς το παρόν υποστηρίζει την ανάπτυξη για iOS, Android, Windows, BlackBerry 10, OS X, Linux, web browsers, Flash, PlayStation 3, PlayStation Vita, Xbox 360, Windows Phone 8, και το Wii U.
Χαρακτηριστικά:
Rendering: Η μηχανή γραφικών χρησιμοποιεί Direct3D (Windows , Xbox 360) , OpenGL (Mac , Windows, Linux) , OpenGL ES (Android , iOS) , και ιδιόκτητες API (κονσόλες). Υπάρχει υποστήριξη για τη ump mapping, reflection mapping, parallax mapping, screen space ambient occlusion (SSAO), dynamic shadows με shadow maps, render-to-texture και full-screen post-processing effects. Υποστηρίζει στοιχεία και μορφές αρχείων από το 3ds Max, Maya, Softimage, Blender, Modo, ZBrush, Cinema 4D, Cheetah3D, Adobe Photoshop, Adobe Fireworks, τα οποία μπορούν να προστεθούν στο προτζεκτ του παιχνιδιού και να διαχειριστούν. Η γλώσσα ShaderLab χρησιμοποιείται για shaders , υποστηρίζοντας τόσο δηλωτική "προγραμματισμού" των προγραμμάτων του αγωγού σταθερής λειτουργίας και shader γραμμένο σε GLSL ή Cg.
Scripting, Asset Tracking, Πολλαπλές πλατφόρμες, Φυσικοί νόμοι.
Παιχνίδια που έχουν δημιουργήθεί: Gone Home (first-person interactive story adventure video game), Game of Thrones: Seven Kingdoms (fantasy MMORPG).

- **JmonkeyEngine** [17]

Φτιάχτηκε ειδικά για τη σύγχρονη τρισδιάστατη ανάπτυξη, καθώς χρησιμοποιεί εκτενώς την τεχνολογία shader. Είναι γραμμένο αποκλειστικά σε Java και χρησιμοποιεί Lightweight Java Game Library ως την προεπιλεγμένη λειτουργία απόδοσης του. Χρησιμοποιείται από διάφορα διαφημιστικά στούντιο παιχνίδιων και τα εκπαιδευτικά ιδρύματα. Το επίσημο jMonkeyEngine 3 SDK είναι ένα περιβάλλον ανάπτυξης παιχνιδιού υψήλοτερου επιπέδου, με πολλαπλά γραφικά στοιχεία. Το SDK βασίζεται στην πλατφόρμα NetBeans, επιτρέποντας επεξεργαστές γραφικών και δυνατότητες plugin.
Παιχνίδια: Chaos (fantasy RPG), Drohtin (RTS) ingleplayer/Multiplayer.

- **IW_Engine** [18]

Αναπτύχθηκε από την Infinity Ward για τη δημιουργία της σειράς παιχνιδιών Call of Duty. Είναι ιδιόκτητη με την συμπερίληψη του προγράμματος GtkRadiant απο την id Software. Έχει χρησιμοποιηθεί από την Infinity Ward, Treyarch, Raven Software και Sledgehammer Games.

- **Blender Game Engine** [19]

Είναι ένα εξάρτημα του Blender, ενός ελεύθερου λογισμικού ανοικτού κώδικα και μίας ολοκληρωμένης σουίτας τρισδιάστατης παραγωγής, που χρησιμοποιείται για την κατασκευή

διαδραστικού περιεχομένου σε πραγματικό χρόνο. Η μηχανή του παιχνιδιού γράφτηκε από το μηδέν σε C++ και περιλαμβάνει υποστήριξη για λειτουργίες όπως η Python scripting και OpenAL 3D ήχο.
Χρησιμοποιεί ένα σύστημα γραφικών logic bricks για να ελέγχει την κίνηση και την απεικόνιση των αντικειμένων στην μηχανή, μπορεί επίσης να επεκταθεί μέσω της Python.
Συμπεριλαμβάνει ανίχνευση σύγκρουσης και προσομοίωση δυναμικής, τύπους σχημάτων, όπως σφαίρα, κουτι, κλπ, πλήρη υποστήριξη για τη δυναμική οχήματος. Υποστηρίζειε όλους τους τρόπους φωτισμού OpenGLTM , συμπεριλαμβανομένων των διαφανειών, αντιστοίχιση υφών, υποστήριξη πολλαπλών υφών, χαρτογραφηση, αναπαραγωγή παιχνιδιών, και τρισδιάστατο ηχο.
Χρησιμοποιήθηκε στην ταινία SpiderMan2 και σε εκπομπές του History Channel.
Παιχνίδια: Yo Frankie! (third person shooter).

- **CryEngine** [20]

Έχει σχεδιαστεί απο την γερμανική εταιρία προγραμματισμού παιχνιδιών Crytek. Η πρώτη έκδοσή της η CryEngine 1, χρησιμοποιήθηκε για το FPS παιχνίδι Far Cry, και το MMORPG, Aion: the tower of eternity. Με την CryEngine2, σχεδιάστηκε το Crysis FPS παιχνίδι. Με την 3 τα Crysis 1,2,3 για Microsoft Windows,PlayStation 3, Xbox 360, και το Warface, αναμένεται η $4^{η}$ γενιά.
Κάποια απο τα χαρακτηριστικά της είναι γράφημα ροής, επεξεργαστής υλικού, ολοκληρωμένο συστημα κάλυψης εδάφους και βλάστησης, εργαλεία για δημιουργία ποταμών και δρόμων, δημιουργία οχημάτων, πλήρως ευέλικτο σύστημα ημέρας, συνεχής ροή, σύστημα οπτικού προυπολογισμού, υπόστήριξη πολυ-πύρηνου επεξεργαστή, απόδοση εκτός σύνδεσης, μεταγλώττιση πόρων, δυναμικός σφαιρικός φωτισμός σε πραγματικό χρόνο, τρισδιάστατο νερό υψηλής ποιότητας, απόδοση υφής σε υψηλή ταχύτητα, δυναμικοί ήχοι και διαδραστική μουσική, ήχοι περιβάλλοντος.
Παιχνίδια: Crysis.

- **3D Studio Max** [21]

Προγραμμα δημιουργίας τρισδιάστατων γραφικών, κινουμένων σχεδίων, μοντέλα, εικόνες. Δημιουργήθηκε και αναπτύχθηκε από την Autodesk Media and Entertainment. Έχει δυνατότητες μοντελοποίησης, μια ευέλικτη αρχιτεκτονική plugin και μπορεί να χρησιμοποιηθεί για την πλατφόρμα Microsoft Windows. Συχνά χρησιμοποιείται από τους σχεδιαστές παιχνιδιών βίντεο, απο τηλεοπτικά στούντιο. Η τελευταία έκδοση του 3ds Max διαθέτει επίσης shaders, δυναμική προσομοίωση, τα συστήματα σωματιδίων, radiosity, κανονική δημιουργία χάρτη και απόδοση καθολικού φωτισμού, ένα προσαρμόσιμο περιβάλλον εργασίας χρήστη, και τη δική του γλώσσα scripting.
Χαρακτηρικτικά: scripting γλώσσα την MAXScript, κίνηση εικονικών χαρακτήρων, εξερεύνηση σκηνής, εισαγωγή DWG αρχείων, προσδιορισμός και επεξεργασία υφών, γενικό keyframing, εξαναγκαστικό animation μεταξύ δυο αντικειμένων, Skinning, σκελετοί και κίνηση (inverse kinematics (IK)), μηχανή αναπαράστασης ένδυσης, ολοκλήρωση με το Autodesk Vault.
Το Avatar χρησιμοποιήσε γραφικά του 3ds_Max.
Χρησιμοποιείται για τρισδιάστατα γραφικά για videogames.

2.4 Fishtank

2.4.1 Περιγραφή

Το παιχνίδι της πτυχιακής εργασίας, αναπαριστά ένα ενυδρείο μέσα στο οποίο υπάρχουν ψάρια. Ο σκοπός του παιχνιδιού είναι να πιάσουμε τα ψάρια και να μαζέψουμε πόντους. Για κάθε ψάρι αντιστοιχεί και ένας πόντος. Υπάρχουν δύο τρόποι ψαρέματος· ο πρώτος είναι με ένα καμάκι που πετάμε στο ψάρι που θέλουμε να πιάσουμε και ο δεύτερος είναι με ένα αγκίστρι που βρίσκεται μέσα στη γυάλα και το οποίο κινούμε ώστε να αγγίξει κάποιο ψάρι.

Σαν εντολές εισόδου, έχει το ποντίκι και το πληκτρολόγιο. Η κάμερα η οποία συνοδεύει το καμάκι κινείται μπροστά και πίσω με τη χρήση των πληκτρων W και S, ενώ στρίβει προς όλες τις κατευθύνσεις με την κίνηση του ποντικιού. Το αγκίστρι το οποίο φαίνεται με μία δεύτερη κάμερα, κινείται με τα «βελάκια» του πληκτρολογίου και με τα A και Z.

Τα προγράμματα που χρησιμοποιήθηκαν για την δημιουργία του παιχνιδιού, είναι το SketchUp και το Blender, ενώ για την επεξεργασία των εικόνων χρησιμοποιήθηκε το Photoshop.

Το παιχνίδι μπορεί να χαρακτηριστεί ως first person shooter, εξ αιτίας του καμακιού που πετάμε, αλλά και ώς ένα Real Time τρισδιάστατο παιχνίδι.

2.4.2 Στόχοι

Αρχικά ο στόχος της εργασίας, ήταν ο σχεδιασμός ενός τρισδιάσταστου μοντέλου με το SketchUp. Στη συνέχεια βλέποντας της δυνατότητες του προγράμματος, αποφασίστηκε να δημιουργηθει το αρχικό μέρος στο SketchUp και το υπόλοιπο σε κάποια άλλη εφαρμογή σχεδίασης τρισδιάστατων γραφικών. Η δεύτερη εφαρμογή που χρησιμοποιήσαμε είναι το Blender και ο λόγος που διαλέχθηκε είναι για να δούμε τις δυνατότητες του στον τρισδιάστατο σχεδιασμό.

Ο σκοπός μέχρι τότε ήταν να φτιαχτεί ένα ενυδρίο, έτσι ώστε να δούμε πόσο ρεαλιστικα θα μπορούσε να αποδοθεί ένας κόσμος κάτω απο το νερό καθώς και η αναπαράσταση των φυσικών καταστάσεων των αντικειμένων.

Στη συνέχεια για να αποκτήσει περισσότερο ενδιαφέρον η εργασία, αποφασίσαμε να εξελίξουμε το τσισδιάστατο μοντέλο σε ενα παιχνίδι και έτσι επιπλέον να έχουμε την ευκαιρία να ανακαλύψουμε τις ικανότητες του Blender τόσο στον σχεδιασμό, όσο και στη δημιουργία παιχνιδιών.

3. ΔΙΑΔΙΚΑΣΙΕΣ ΑΝΑΠΤΥΞΗΣ

3.1 Εισαγωγή

Αρχικά χρησιμοποιήθηκε το SketchUp, για την εύκολη χρήση του στο σχεδιασμό τρισδιάστατων μοντέλων και για να δούμε τις δυνατότητές του. Καθώς όμως είναι καθαρά ένα πρόγραμμα για αρχιτεκτονικό σχεδιασμό, αποφασίστηκε να το συνδυάσουμε και με ένα δεύτερο πρόγραμμα.

Το δεύτερο πρόγραμμα είναι το Blender, το οποίο είναι κατάλληλο για σχεδιασμό τρισδιάστατων μοντέλων, αλλά και για τη δημιουργία διαδραστικών εφαρμογών. Επίσης θέλαμε να δούμε τις δυνατότητες της εφαρμογής αυτής καθώς και την ευκολία στη χρήση και την κατανόησή της.

3.2 Εργαλεία

3.2.1 SketchUp

3.2.1.1 Εισαγωγή [22]

Το SketchUp είναι ένα ελεύθερο λογισμικό για τρισδιάστατο σχεδιασμό αντικειμένων. Χρησιμοποιείται κυρίως για αρχιτεκτονικά και μηχανικά σχέδια. Έγινε ιδιοκτησία της Google από το 2006-2012 και από το 2012 μέχρι και σήμερα το έχει αναλάβει η εταιρεία Trimble Navigation, η οποία ασχολείται με χαρτογράφηση, χωρομέτρηση και εξοπλισμούς πλοήγησης.

Το πρόγραμμα είναι εύχρηστο και διαθέτει μία ηλεκτρονική αποθήκη την 3D Warehouse [23] η οποία έχει έτοιμα τρισδιάστατα μοντέλα έτσι ώστε ο κάθε χρήστης να μπορεί να τα κατεβάσει και να τα χρησιμοποιήσει. Επίσης τα μοντέλα που δημιουργούνται στο SketchUp, μπορούν να τοποθετηθούν στο Google Earth.

Οταν το SketchUp άνηκε στην Google, δημιουργήθηκε μια έκδοση του, η οποία ήταν δωρεάν και μπορούσε να την κατεβάσει κάθε χρήστης. Διέθετε ολοκληρωμένα εργαλεία ώστε να ανεβάζει περιεχόμενα στο Google Earth και στο Google 3D Warehouse. Δημιουργήθηκε μια νέα εργαλειοθήκη ώστε ο χρήστης να μπορεί να βλέπει το περιβάλλον του SketchUp από τη σκοπιά του πραγματικού κόσμου. Επίσης η δωρεάν έκδοση μπορεί να εξάγει αρχεία με τη μορφή των .dae, .kmz, .3ds, .dwg, .dxf, .fbx, .obj, .xsi, and .wrl αρχείων. Μπορεί να κάνει render εικόνες με τις καταλήξεις .bmp, .png, .jpg,.tif.

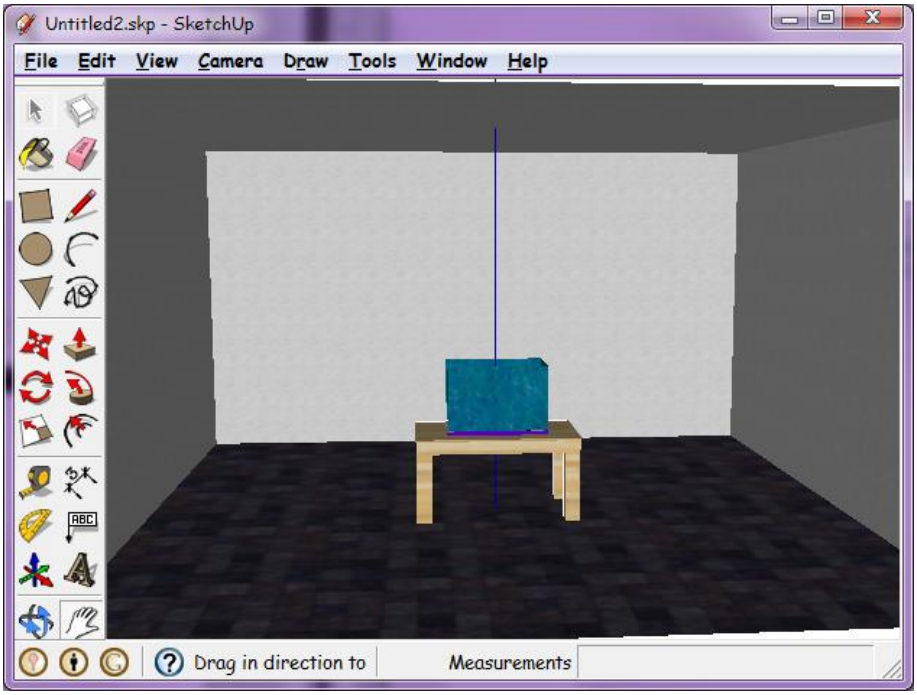
Υπο την Trimble δεν υπάρχουν διαφορές για την δωρεάν έκδοση του SketchUp. Το SketchUp Pro αποκτά παραπάνω δυνατότητες όπως καλύτερα εργαλεία και δυνατότητα εξαγωγής βίντεο υψηλής ανάλυσης.

Το SketchUp έχει εφαρμογή στην αρχιτεκτονική, μηχανολογία, στη δημιουργία φιλμ, τον σχεδιασμό παιχνιδιών και στην τρισδιάστατη εκτύπωση.

3.2.1.2 Σχεδιασμός βασικού σχεδίου [41]

Τα αντικείμενα που φτιάξαμε με το SketchUp είναι το τραπέζι, το ενυδρείο και η βάση του και το δωμάτιο όπου υπάρχουν όλα αυτά. Για τη δημιουργία τους χρησιμοποιήσαμε τα εργαλεία σχεδιασμού και τροποποίησης που διαθέτει το SketchUp [41]. Τους προσθέσαμε διάφορα materials και styles, που διαθέτει η εφαρμογή και κάναμε δοκιμές για το πως ταιριάζουν πάνω στα αντικείμενα [42].

Το τελικό αποτέλεσμα είναι αυτό της εικόνας.



Εικόνα 1 - Τελικό αποτέλεσμα απο το SketchUp

3.2.2 Import – Export

3.2.2.1 Εισαγωγή

Για να μπορέσουμε να συνεχίσουμε την επεξεργασία του μοντέλου στο Blender, πρέπει να το εξάγουμε απο το SketchUp σε μία μορφή αρχείου η οποία να χρησιμοποιείται και απο τα δύο προγράμματα. Η μορφή αυτή είναι η επέκταση αρχείου .dae, η οποία ταυτίζεται με τα αρχεία Collada.

Όπως φαίνεται στο παράρτημα B.3 κάνουμε δοκιμές πάνω στο μοντέλο για δούμε με ποιές υφές και στύλ μπορούμε να έχουμε το καλύτερο αποτέλεσμα όταν το εισάγουμε στο Blender.

3.2.2.2 Collada [24]

Το Collada (Collaborative Design Activity) είναι μια συνεργατική δραστηριότητα σχεδιασμού για τη θέσπιση μιας μορφής ανταλλαγής αρχείων για διαδραστικές τρισδιάστατες εφαρμογές. Διαχειρίζεται από τον όμιλο Khronos [25] μια μη κερδοσκοπική βιομηχανία τεχνολογίας.

Η Collada ορίζει ένα ανοιχτό πρότυπο σχήμα XML για την ανταλλαγή ψηφιακών στοιχείων μεταξύ των διαφόρων λογισμικών εφαρμογής γραφικών, που μπορεί να αποθηκεύσει τα άλλα τους στοιχεία σε μη συμβατές μορφές αρχείων. Τα έγγραφα Collada που περιγράφουν ψηφιακά στοιχεία, είναι αρχεία XML και συνήθως ταυτίζονται με την επέκταση αρχείου .dae (digital asset exchange).

Είναι το πρώτο λογισμικό ανεξαρτήτου πλατφόρμας, το οποίο ορίζει πρότυπα σκίασης και εφέ γραμμένα σε XML. Απευθύνεται σε συστήματα τα οποία τρέχουν OpenGL Shading Language [30] αλλά και σε συστήματα περιορισμένων πόρων. [26]

Η τεχνολογία Collada επικεντρώνεται στον τομέα της ψυχαγωγίας, όπου το περιεχόμενο είναι τρισδιάστατο και συνήθως είναι ένα παιχνίδι. Υποστηρίζει όλα τα χαρακτηριστικά που χρειάζονται οι νέες τρισδιάστατες εφαρμογές για να ανταλλάξουν και να διαφυλάξουν πλήρως τα στοιχεία τους. Οι δυνατότητες της επεκτείνονται ώστε να ενσωματώσει τεχνολογίες, όπως τα προγραμματιζόμενα εφέ σκιάς και τον έλεγχο μηχανών φυσικής πραγματικού χρόνου.

Ωστόσο δεν είναι απλά μια τεχνολογία. Έχει καταφέρει να προσφέρει μια ουδέτερη ζώνη, όπου οι ανταγωνιστικές εταιρείες συνεργάζονται για την ανάπτυξη κοινών προδιαγραφών. Μπορεί επίσης να επεκταθεί από τους τελικούς χρήστες ανάλογα με τους σκοπούς που θέλουν να πετύχουν.

Η επιλογή της XML σαν γλώσσα της Collada, έγινε για να εκμεταλλευτούν τα πλεονεκτήματα της, συμπεριλαμβανομένης και της διεθνούς κωδικοποίησης UTF-8.

Το Collada υποστηρίζεται από διάφορα λογισμικά δημιουργίας δισδιάστατων ή τρισδιάστατων γραφικών, καθώς και απο μηχανές παιχνιδιών. Μερικά απο αυτά είναι:

- Photoshop της Adobe
- 3DS Max της Autodesk
- Maya της Autodesk
- AutoCAD της Autodesk
- SketchUp της Trimble Navigation
- Google Earth της Google
- Blender της Blender Foundation
- Unity της Unity Technologies
- Panda 3D της Disney
- Cry Engine 2 της Crytek
- Unreal Engine της Epic Games

3.2.2.3 Συμπεράσματα

Απο τις δοκιμές που έχουμε κάνει στο παράρτημα [47], βλέπουμε ότι τα χρώματα που βάζουμε στο μοντέλο από το SketchUp εμφανίζονται και στο Blender και στο Solid και στο Texture Mode, ενώ τα Textures που βάζουμε εμφανίζονται μόνο στο Texture Mode σωστά. Η γεωμετρία και οι ακμές εμφανίζονται και στο Blender. Οι ακμές μάλιστα είναι η μόνη επιλογή από τα Edge Styles η οποία φαίνεται στο Blender. Απο τις επιλογές Face Style καμία δεν μεταφέρεται κατά την εξαγωγή, όπως και οι επιλογές Shadow και Fog.

Μετά από τις δοκιμές που έγιναν παραπάνω, ο ποιός εύκολος τρόπος να χειριστούμε το μοντέλο με το Blender είναι το κάνουμε Export από το SketchUp, βάζοντας μόνο υφή στη γυάλα και στους τοίχους του "δωματίου" έτσι ώστε να μπορούμε να δουλέψουμε με τα βασικά αντικείμενα ολόκληρα και όχι διασπασμένα σε υπο-αντικείμενα.

3.2.3 Blender

3.2.3.1 Εισαγωγή [27]

Το Blender είναι ένα ελεύθερο λογισμικό ανοικτού κώδικα, για τρισδιάστατα γραφικά υπολογιστή. Χρησιμοποιείται για τη δημιουργία ταινιών με κινούμενα σχέδια, οπτικά εφέ, τρισδιάστατη εκτύπωση μοντέλων, διαδραστικων τρισδιάστατων εφαρμογών και βίντεο-παιχνιδιών. Στα χαρακτηριστικά του περιλαμβάνονται ο τρισδιάστατος σχεδιασμός, το UV unwrapping, η δημιουργία υφών, αναπαράσταση καπνού, υγρών, σωματιδίων και μαλακών αντικειμένων, γλυπτική, κίνηση, κάμερα παρακολούθησης, απόδοση σε πραγματικό κόσμο, επεξεργασία και σύνθεση βίντεο. Επίσης διαθέτει ενσωματωμένη μηχανή δημιουργίας παιχνιδιών καθώς και τη γλώσσα προγραμματισμού Python για εσωτερικό scripting.

Είναι διαθέσιμο για όλα τα κύρια λειτουργικά συστήματα όπως τα Windows της Microsoft, τα Linux και το Mac OS X.

Έχει χρησιμοποιηθεί αρχικά για κινούμενα σχέδια και προ-απεικονίσεις για την ταινία Spider-Man 2. Επίσης για εκπομπές του History Channel, καθώς και για διαφημηστικά τηλεόρασης σε χώρες όπως η Αυστραλία, Ισλανδία, Βραζιλία, Ρωσσία και Σουηδία.

Σχεδόν κάθε χρόνο το ίδρυμα Blender ανακοινώνει κάποιο καινούριο project. Το 2006 παρουσιάστηκε η ταινία κινουμένων σχεδίων μικρού μήκους Elephants Dream, το 2008 η Big Buck Bunny, το 2010 η Sintel και το 2012 η Tears of Steel. Ενώ το 2008 δημιουργήθηκε και ανακοινώθηκε το τρισδιάστατο παιχνίδι Yo Frankie.

3.2.3.2 Χρήση

Αφού εισάγουμε το μοντέλο στο Blender [45], ξεκινάμε να το επεξεργαζόμαστε πλέον σαν παιχνίδι [54], χρησιμοποιώντας την μηχανή Game Engine που διαθέτει.

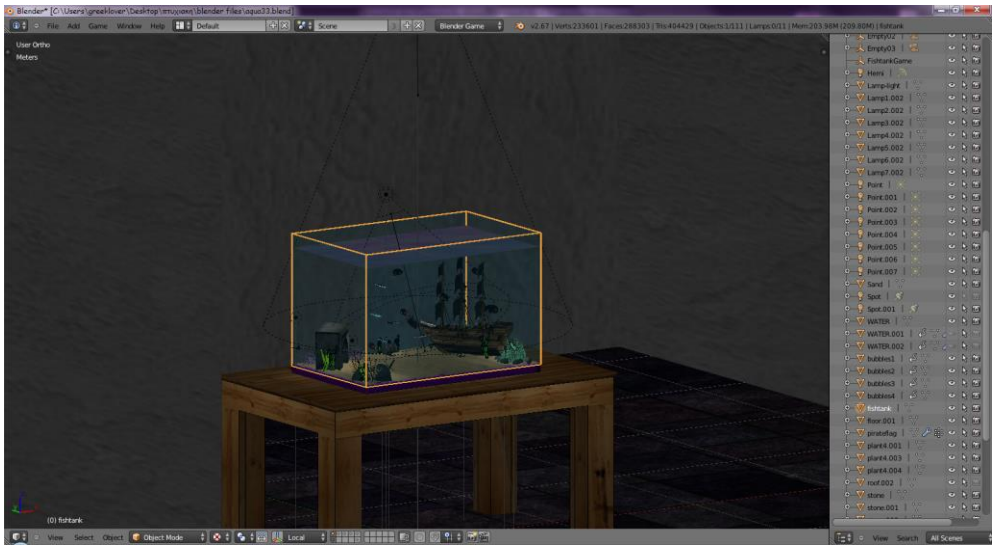
Δημιουργούμε με τα εργαλεία του Blender [52], τα έξτρα αντικείμενα που θα χρειαστούν για τη δημιουργία του περιβάλλοντος μέσα στο ενυδρίο [56], όπως το στρώμα της άμμου, την επιφάνεια του νερού, το αγκίστρι, τα φύκια και το καμάκι. Κάποια άλλα αντικείμενα όπως τα ψάρια, και το πλοίο, τα έχουμε κατεβάσει απο την σελίδα Turbosquid [28]. Επίσης έχουμε χρησιμοποιήσει λάμπες για το φωτισμό της σκηνής και για να την κάνουμε πιο ρεαλιστική, καθώς και κάμερες για την πλοήγηση μέσα στο ενυδρίο.

Με τα εργαλεία Materials και Textures δίνουμε στα αντικείμενα διάφορες ιδιότητες όπως χρώμα, φωτεινότητα και επίσης τους δίνουμε υφή τοποθετώντας πάνω τους εικόνες, με τέτοιο τρόπο, ώστε να φαίνεται σαν τη φυσική επιφάνειά τους.

Στα αντικείμενα που αναπαριστούν τα ψάρια, όπως και στα φύκια για να τα κάνουμε να φαίνονται πιο πραγματικά, τους προσθέσαμε κίνηση με τη χρήση σκελετού.

Αναλυτικά ο τρόπος που φτιάχνονται όλα τα παραπάνω ειναι στο παράρτημα [52].

Το αποτέλεσμα θα ειναι αυτό της εικόνας.



Εικόνα 2 - Τελικό αποτέλεσμα απο το Blender

Αφού τελειώσουμε με το στήσιμο της σκηνής, πρέπει να διαμορφώσουμε το μοντέλο μας, ώστε να συμπεριφέρεται σαν παιχνίδι. Τον ρόλο αυτό τον αναλαμβάνει ο Game Logic Editor, ο οποίος αναθέτει σε κάθε αντικείμενο εντολές. Επίσης μπορούμε να χρησιμοποιήσουμε script της γλώσσας προγραμματισμού Python αν θέλουμε κάποιο αντικείμενο να ακολουθήσει πιο συγκεκριμένες εντολες. Για παράδειγμα για την κίνηση της κάμερας και του αγκιστριού έχουν χρησιμοποιηθεί δύο διαφορετικά script. Τα script τοποθετούνται πάνω σε ένα αντικείμενο σε συνδυασμό με τον Game Logic Editor.

Για το μέτρημα των πόντων χρησιμοποιήσαμε τα Texts τα οποία τοποθετήσαμε σε μία δεύτερη σκηνή την οποία ρυθμίσαμε να φαίνεται ταυτόχρονα με την βασική σκηνή μας.

Προσθέσαμε ήχο, ο οποίος αναπαράγεται κατά τη διάρκεια του παιχνιδιού, τον οποίο τον ρυθμίσαμε μαζί με το σκορ του παιχνιδιού σε ένα κοινό αντικείμενο.

Για τα αντικείμενα τα οποία αναπαριστούν τις φυσαλίδες, τους βάλαμε μια φυσική κατάσταση, ώστε να ανεβαίνουν προς τα πάνω.

3.3 Διαδικασίες μοντελοποιήσης

3.3.1 Εισαγωγή

Η βασική μοντελοποιήση του παιχνιδιού, έγινε με την εφαρμογή Blender. Στο μοντέλο του παιχνιδιού, προστέθηκαν textures, physics και bones, ενώ για τη δράση του χρησιμοποιήθηκαν διάφορα actions απο την game engine του Blender.

Μερικά απο τα βασικά εργαλεία σχεδίασης είναι τα Meshes, Empties, Lamps,Cameras, Bézier Curves, Texts, Armatures.

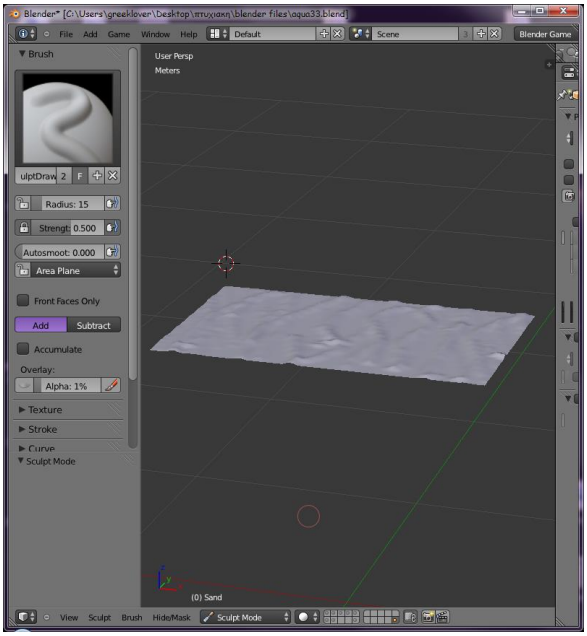
3.3.1.1 3D Cursor

Ο 3D Cursor χρησιμοποιείται για την τοποθέτηση ενός νέου αντικειμένου. Ο προεπιλεγμένος τρόπος για εύκολη τοποθέτησή του είναι με το αριστερό κλικ του ποντικιού, πάνω στη σκηνή. Παρόλα αυτά εμείς χρησιμοποιούμε δύο hotkeys για μεγαλύτερη ακρίβεια. Το πρώτο είναι το shift C με το οποίο τοποθετούμε τον κέρσορα στο κέντρο των αξόνων της σκηνής. Το δεύτερο είναι το shift S, το οποίο μας προσφέρει διάφορες επιλογές. Την θέση του κέρσορα μπορούμε να τη ρυθμίσουμε επίσης από την καρτέλα Properties που βρίσκεται κρυμμένη στα δεξιά της σκηνής.

3.3.1.2 Mesh

Είναι τα έτοιμα πλέγματα αντικειμένων που διαθέτει το Blender, τα οποία μπορούμε να χρησιμοποιήσουμε μόνα τους ή σε συνδυασμούς ώστε να δημιουργήσουμε απλά ή σύνθετα αντικείμενα. Με αυτόν τον τρόπο φτιάξαμε το αντικείμενο για την άμμο του ενυδρίου [56], την

επιφάνεια του νερού. Το στρώμα της άμμου το κάναμε ανάγλυφο με τη βοήθεια του Sculpt Mode που διαθέτει το Blender.



Εικόνα 3 – Mesh-Plane για το στρώμα της άμμου

3.3.1.3 Empty [29]

Το "Empty" είναι ένα κενό αντικείμενο που δεν περιέχει καμία γεωμετρία. Μπορεί να χρησιμεύσει ως χειριστής άλλων αντικειμένων.

Μερικά παραδείγματα των τρόπων χρήσης τους:

- Μπορεί να γίνει γονικό αντικείμενο για ένα ή περισσότερα αντικείμενα, δίνοντας έτσι στο χρήστη τη δυνατότητα να ελέγχει ευκολότερα πολλά αντικείμενα, χωρίς να επηρεάζεται η αποδόση ή εκκίνηση του παιχνιδιού
- Μπορεί να χρησιμοποιηθεί για την φυσική κατάσταση των αντικειμένων, ή για τις ανάγκες του οπλισμού, δίνοντας στο χρήστη πολύ περισσότερο έλεγχο
- Μπορεί να χρησιμοποιηθεί για να αντισταθμίσει πολύπλοκες παραμορφώσεις

Στο παιχνίδι το Empty έχει δύο χρήσεις, σαν parent για τις κάμερες και σαν το αντικείμενο στο οποίο έχουν ανατεθεί, οι ήχοι που θα αναπαράγονται στο παιχνίδι και το σκορ.

3.3.1.4 Camera [30]

Η camera είναι ένα αντικείμενο που παρέχει εικόνες που έχουν αποδοθεί απο το Blender. Στην Game Engine λειτουργεί ως συσκευή σε πραγματικό χρόνο, έχοντας μια σειρά απο πρόσθετα χαρακτηριστικά· μπορεί να χρησιμοποιηθεί ως στατική κάμερα, αλλά και ως μια κινούμενη συσκευή με τα προκαθορισμένα της χαρακτηριστικά.

Επίσης οποιοδήποτε αντικείμενο στο παιχνίδι μπορεί να χρησιμοποιηθεί σαν κάμερα, ανάλογα με τον προσανατολισμό του.

Καθορίζει ποιά τμήματα της σκηνής θα είναι ορατά κατά τον χρόνο απόδοσης.

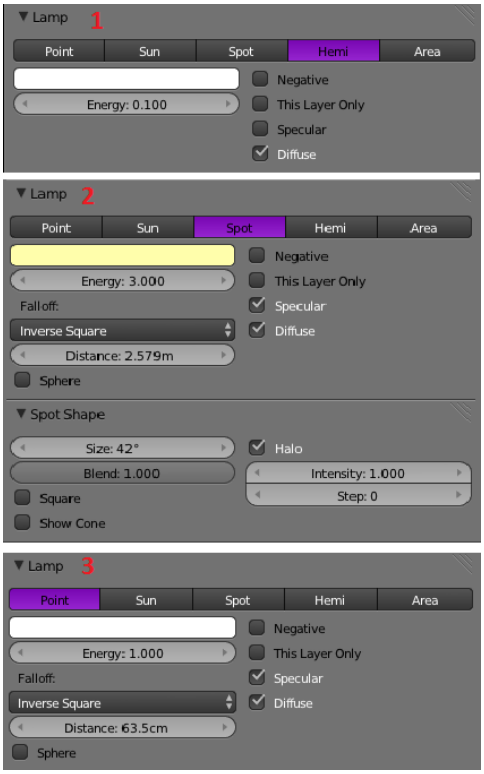
3.3.1.5 Lamp [31]

Για το φωτισμό της σκηνής, χρησιμοποιούμε τα Lamps. Απο το μενού μπορούμε να διαλέξουμε ανάλογα με το τι μας εξυπηρετεί, ένα από τα 5 είδη που έχει το Blender.

Για τη σκηνή του παιχνιδιού, χρησιμοποιήσαμε σαν βασικό φώς το Hemi το οποίο φωτίζει παντού ανεξάρτητα από τα αντικείμενα που στέκονται εμπόδια στο φως.

Με το Spot, που είναι σαν μικρός προβολέας, φωτίζουμε το Fishtank ξεχωριστά από την υπόλοιπη σκηνή.

Με το Point φωτίζουμε μεμονωμένα αντικείμενα, και το έχουμε βάλει σε διάφορα Objects μέσα στο Fishtank.



Εικόνα 4 - Properties Editor για τα lamps

3.3.1.6 Text [32]

Τα Texts είναι αντικείμενα, όπως τα Béziers curves τα οποία περιέχουν κείμενο. Επιτρέπουν να δημιουργηθεί δισδιάστατο ή τρισδιάστατο κείμενο, με δυνατότητα μετακίνησης, περιστροφής και όλων των ιδιοτήτων που έχουν τα υπόλοιπα αντικείμενα. Σε Edit Mode μπορούμε να αλλάξουμε το περιεχόμενο του κειμένου. Με alt C σε Object Mode μπορούμε να μετατρέψουμε ένα Text σε Mesh.

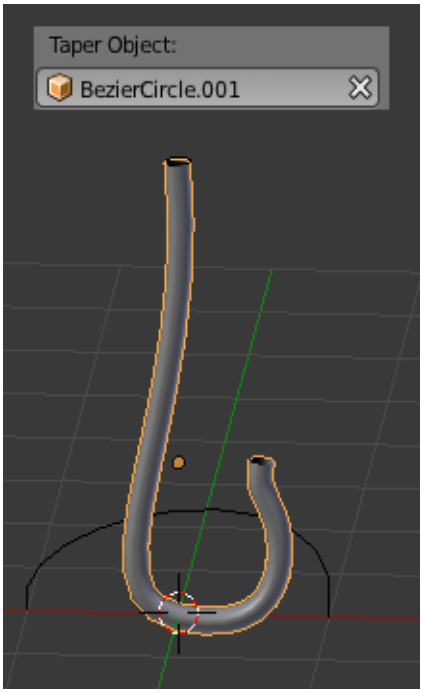
Στο παιχνίδι τα χρησιμοποιούμε για να φτιάξουμε τα αντικείμενα για το μέτρημα των πόντων, όπως φαίνεται και στο παράρτημα [80].

3.3.1.7 Bézier Curve [33]

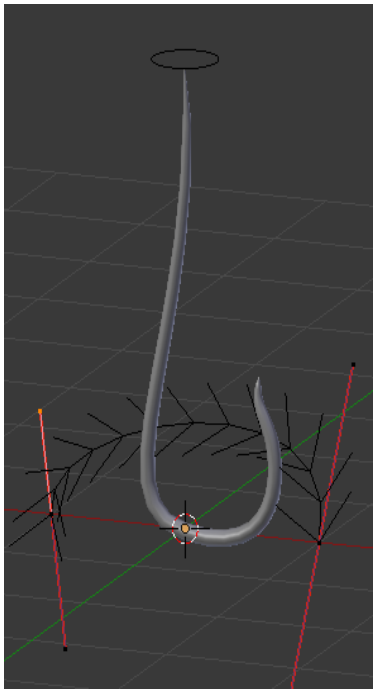
Μία καμπύλη Bézier είναι μια παραμετρική καμπύλη που χρησιμοποιείται σε γραφικά υπολογιστών. Σε διανυσματικά γραφικά, χρησιμοποιούνται για την μοντελοποίηση απαλών καμπυλών. Συνήθως αναφέρονται ως "Διαδρομές", όπως σε προγράμματα επεξεργασίας εικόνας, όπου είναι συνδυασμοί των συνδεδεμένων καμπυλών Bézier. Χρησιμοποιούνται επίσης σε animation ως εργαλείο για τον έλεγχο της κίνησης. [34]

Σε 3D εφαρμογές, χρησιμοποιούνται για να καθορίσουν τρισδιάστατα μονοπάτια.

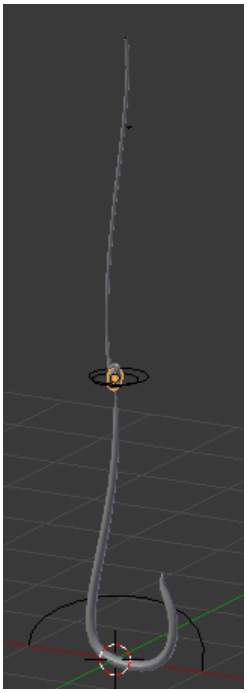
Με τη βοήθεια των καμπυλών Bézier φτιάξαμε τον "παίχτη" του παιχνιδιού μας, ένα αγκίστρι και κάποια αντικείμενα τα οποία έχουν τη μορφή των φυκιών[57].



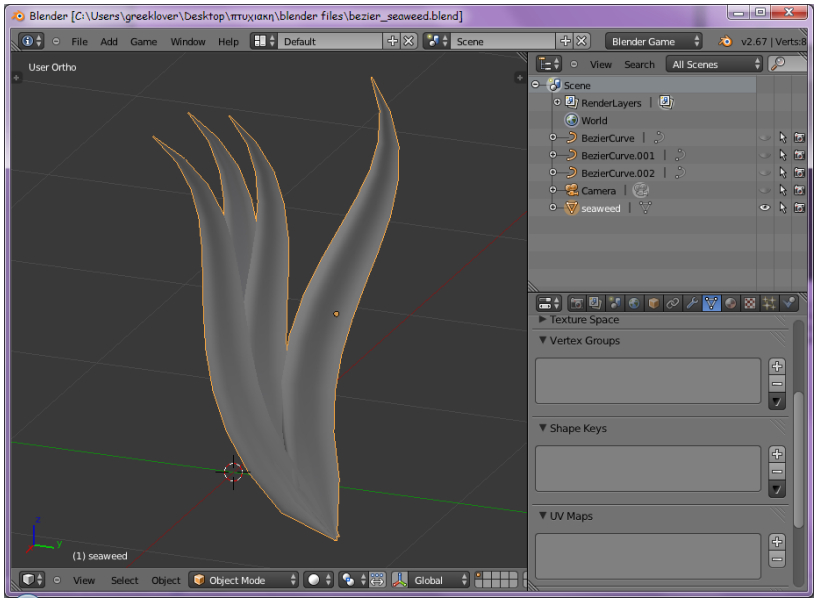
Εικόνα 5 - διαμόρφωση 1



Εικόνα 6 – διαμόρφωση 2



Εικόνα 7 - τελικό αποτέλεσμα



Εικόνα 8 – φύκια φτιαγμένα με Béziers curves

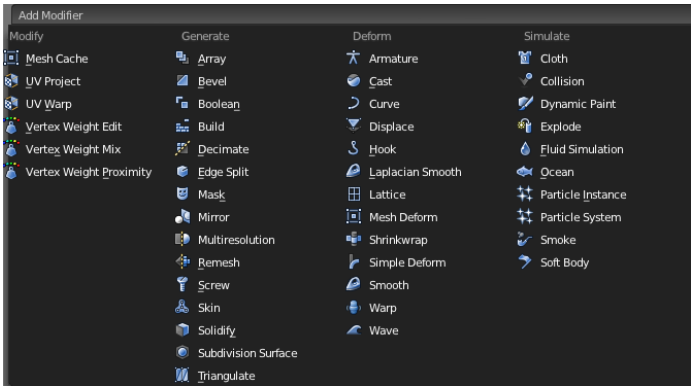
3.3.1.8 Object Modifier [35]

Οι Modifiers είναι λειτουργίες που εκτελούνται αυτόματα και επηρεάζουν ένα αντικείμενο με χωρίς όμως να καταστρέφεται η αρχική του δομή και γεωμετρία. Με τη χρήση τους μπορούν να εκτελεστούν μηχανικά πολλές ενέργειες που θα ήταν κουραστικό να γίνουν με το χέρι και χωρίς να αλλάζει η βασική γεωμετρία του αντικειμένου. Η χρηστικότητά τους είναι στο ότι αλλάζουν τον τρόπο με τον οποίο ένα αντικείμενο εμφανίζεται και αποδίδεται, αλλά μόνο εικονικά. Μπορούν να προστεθούν πολλοί Modifiers σε ένα αντικείμενο. Με το Apply τους εφαρμόζουμε μόνιμα σε αυτό.

Στην παρακάτω εικόνα έχουμε τους Modifiers, τους οποίους μπορούμε να χρησιμοποιήσουμε. Υπάρχουν 4 τύποι Modifier.

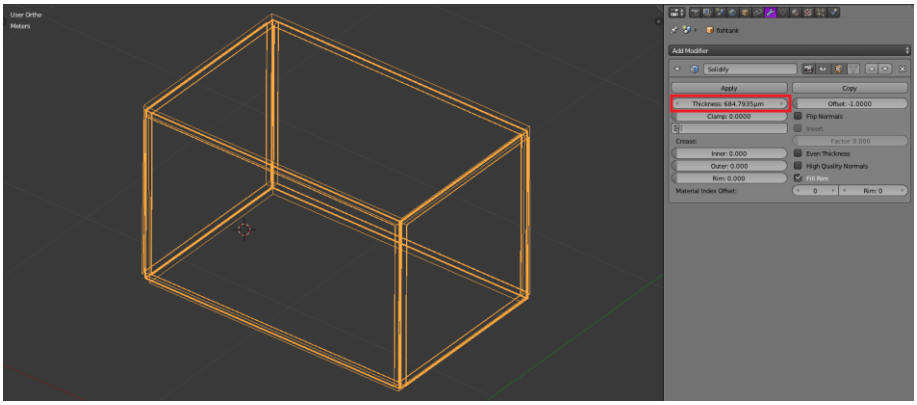
- **Modify**: δεν επηρεάζουν άμεσα το σχήμα του αντικειμένου. Σε αυτό ανήκει το UV Project το οποίο βάζει UV συντεταγμένες στο αντικείμενο και το UV Warp το οποίο επεξεργάζεται δυναμικά τις UV συντεταγμένες

- **Generate**: αλλάζουν τη γενική εμφάνιση του αντικειμένου, ή του προσθέτουν αυτόματα νέα γεωμετρία. Του ανήκουν Modifiers, όπως ο πολλαπλασιασμός του αντικειμένου συμμετρικά, με το Mirror, το Solidify, όπου δίνει βάθος στα αντικείμενα
- **Deform**: αλλάζουν το σχήμα ενός αντικειμένου όπως το Armature, το Smooth, Wave
- **Simulate**: αυτοί οι Modifiers προστίθενται αυτόματα, όταν είναι ενεργοποιημένο, ένα Physics simulation. Δημιουργούν στα αντικείμενα μία φυσική ιδιότητα. Στο Simulate, ανήκουν οι εξομοιώσεις ρούχων (Cloth), υγρών (Fluid), καπνού (Smoke), ωκεανού (Ocean)



Εικόνα 9 - Modifiers

Στο Fishtank χρησιμοποιήσαμε τον Solidify Modifier για να δώσουμε πάχος στα τοιχώματα της γυάλας. Όπως φαίνεται και στην εικόνα 10, όσο αυξάνουμε το Thickness, τόσο πιο παχύ γίνεται το object. Πατώντας Apply, εφαρμόζουμε μόνιμα την αλλαγή πάνω στο object.



Εικόνα 10 – εφαρμογή του Solidify Modifier

3.3.1.9 Materials and Textures

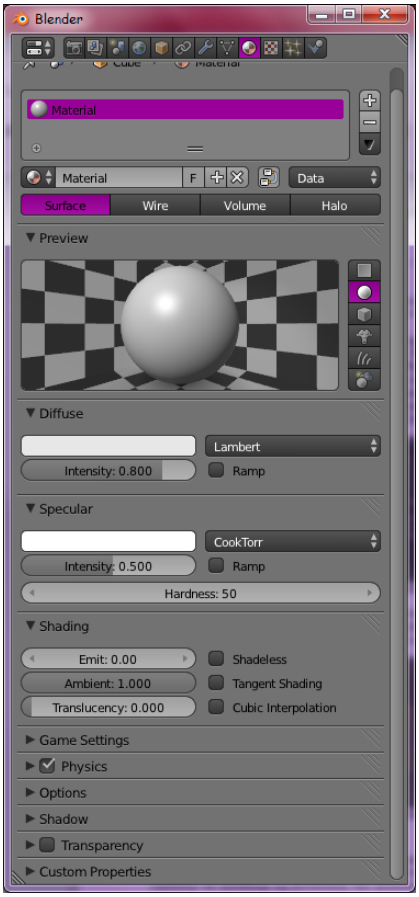
Δύο από τα σημαντικότερα εργαλεία του Blender είναι τα Materials και Textures. Με τη χρήση τους δίνουμε στα αντικείμενα ξεχωριστές ιδιότητες και τα κάνουμε να φαίνονται ρεαλιστικά. Παρακάτω θα εξετάσουμε τις δυνατότητες τως εργαλείων αυτων για την μηχανή Game Engine του Blender.

Materials [36] [37] [38]

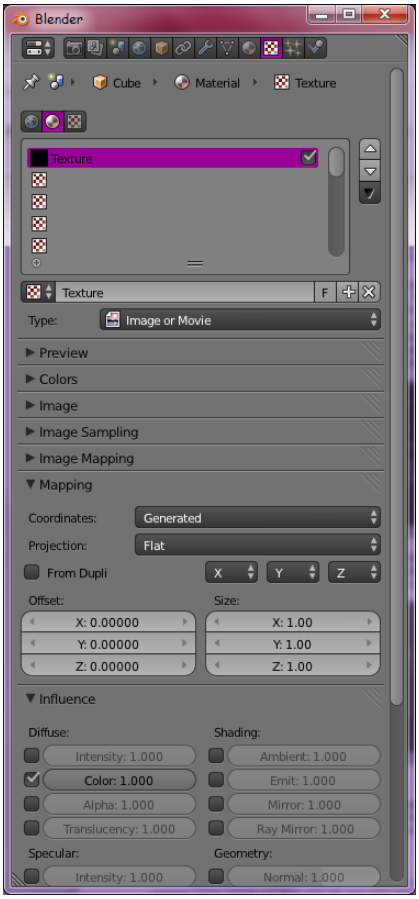
Με το εργαλείο material, ορίζουμε τις ιδιότητες που θέλουμε να έχει το αντικείμενο μας κατά την απόδοσή του. Οι ιδιότητες αυτές μπορεί να είναι από την πιο απλή όπως το χρώμα του αντικειμένου, έως και πιο σύνθετες όπως ο τρόπος που θα παρουσιάζεται κατά την εκκίνηση του παιχνιδιού.

Σε κάθε αντικείμενο μπορούμε να βάλουμε όσα Material θέλουμε. Μπορούμε να χρησιμοποιούμε όποιο μας βολεύει κάθε φορά, ή να χρησιμοποιήσουμε πάνω απο ένα ταυτόχρονα.

Για κάθε αντικείμενο μπορούμε να ορίσουμε παραπάνω απο ένα materials. Για κάθε material που ορίζουμε στο αντικείμενο, μπορούμε να φτιάξουμε textures, απο τα οποία μόνο ένα μπορεί να χρησιμοποιηθεί[43].



Εικόνα 11 – καρτέλα για τα Materials



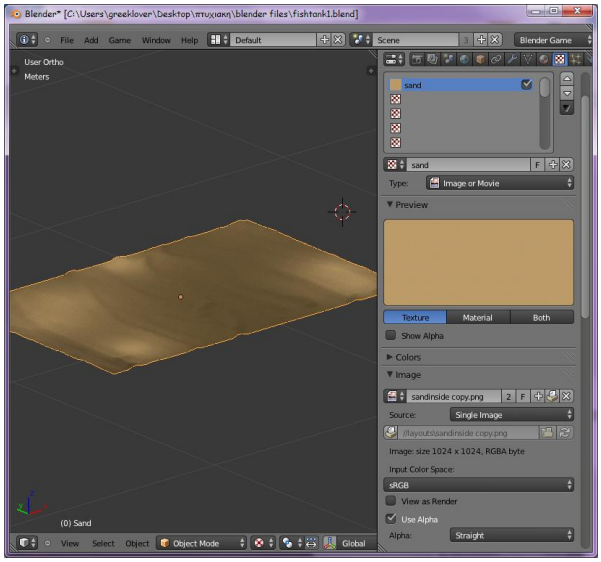
Εικόνα 12 – καρτέλα για τα Textures

Textures [39] [40]

Με τα Textures προσθέτουμε υφές σε ένα αντικείμενο. Μπορούμε να προσθέσουμε έτοιμες εικόνες ή εικόνες που έχουν επεξεργαστεί στο Photoshop πάνω σε ένα αντικείμενο, με βάση τη γεωμετρία του και να φαίνεται πιο ρεαλιστικό [62].

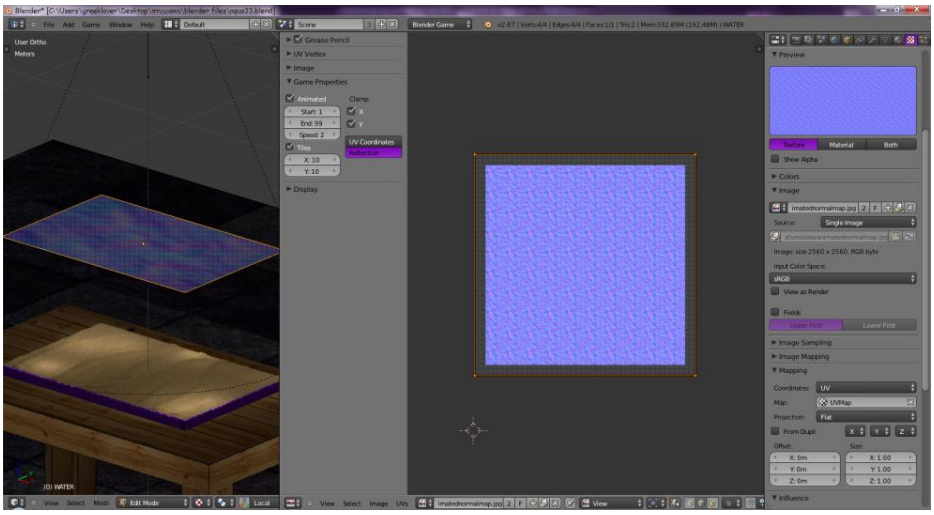
Η πιο αποτελεσματική διαδικασία του texturing είναι η UV Mapping [63] ή αλλιώς χαρτογράφηση. Με αυτή τη μέθοδο, οι επιφάνειες του τρισδιάστατου αντικείμενου «ξεδιπλώνονται» και γίνονται δισδιάστατες, έτσι ώστε να μπορούμε να τις χειριστούμε σαν εικόνα. Μετά την επεξεργασία της εικόνας, ντύνουμε με αυτήν ξανα το αντικείμενο, το οποίο έχει την υφή που θέλουμε, σωστά τοποθετημένη πάνω του.

Στην εικόνα 13 έχουμε τα αποτελέσματα του texturing στο αντικείμενο της άμμου.



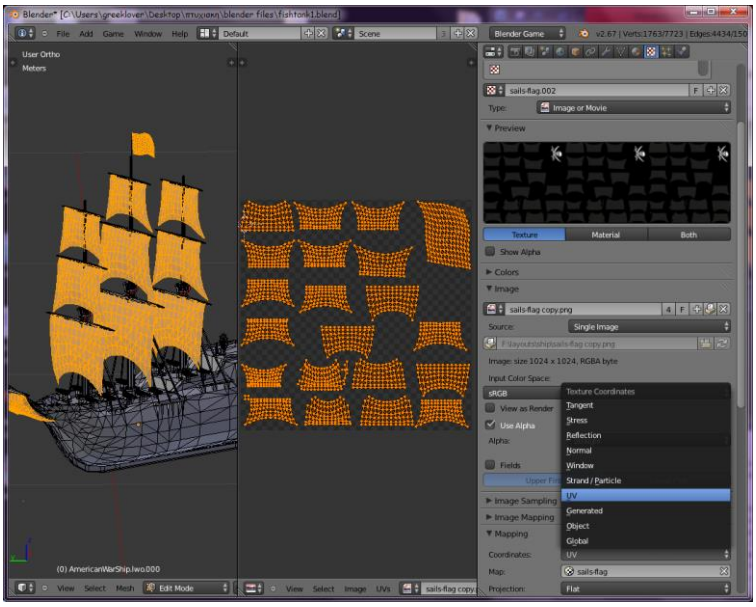
Εικόνα 13 – το Plane της άμμου με Texture

Στην επιφάνεια του νερού, διαλέξαμε μία εικόνα η οποία αποτελείται απο πολλές εικόνες, για να μπορέσουμε να πετύχουμε την υφή του νερού καλύτερα. Εκτός απο την τοποθέτηση της εικόνας πάνω στο αντικείμενο, κάναμε και κάποιες ρυθμίσεις για να δημιουργήσουμε την αίσθηση της κίνησης του νερού [62].

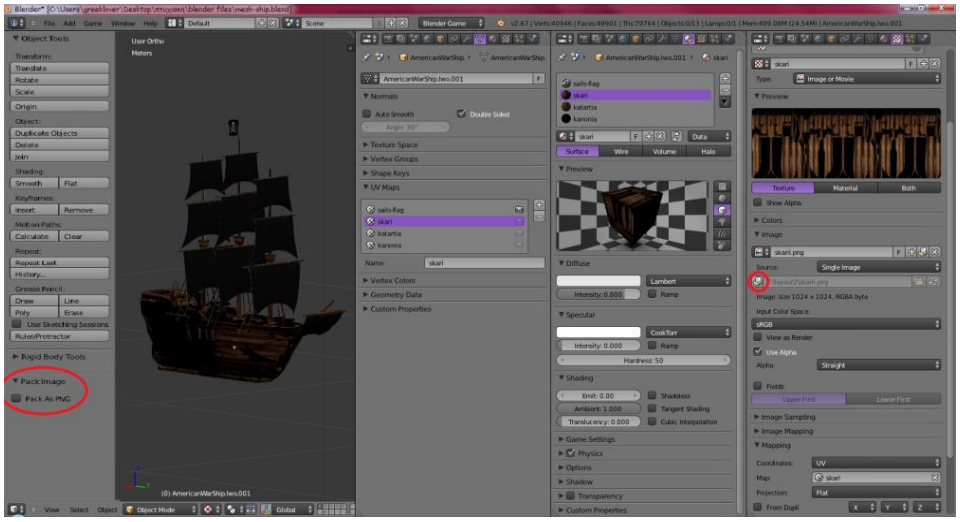


Εικόνα 14 – Texture και Animation ρυθμίσεις για το Plane του νερού

Για το αντικείμενο του πλοίου, χωρίσαμε τα διάφορα τμήματά του, σε διαφορετικά materials, και σε κάθε ένα βάλαμε και απο ένα texture. Επίσης χρησιμοποιήσαμε τη UV Mapping, καθώς οι επιφάνειες του πλοίου ήταν πολύπλοκες.



Εικόνα 15 – Texturing για το Mesh του πλοίου



Εικόνα 16 – όλοι οι Editors που χρησιμοποιήθηκαν για το mesh του πλοίου

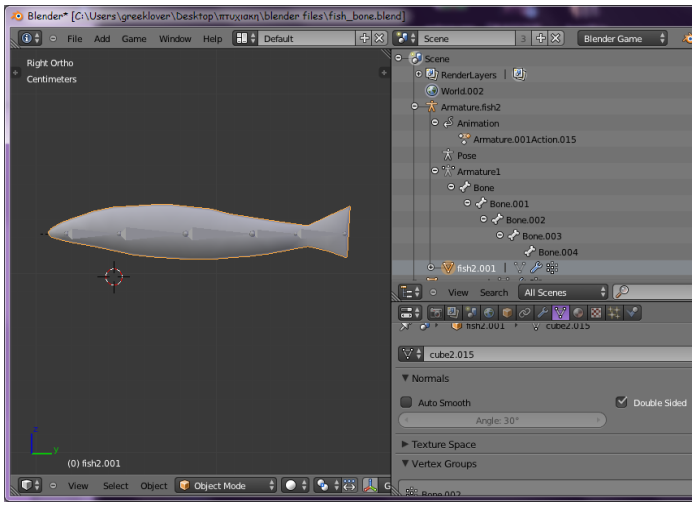
Με αυτόν τον τρόπο βάλαμε υφές σε όλα τα αντικείμενα.

Εικόνα 17 – τελικό αποτέλεσμα σε όλα τα αντικείμενα του μοντέλου

3.3.1.10 Rigging [41] [42] [43] [44]

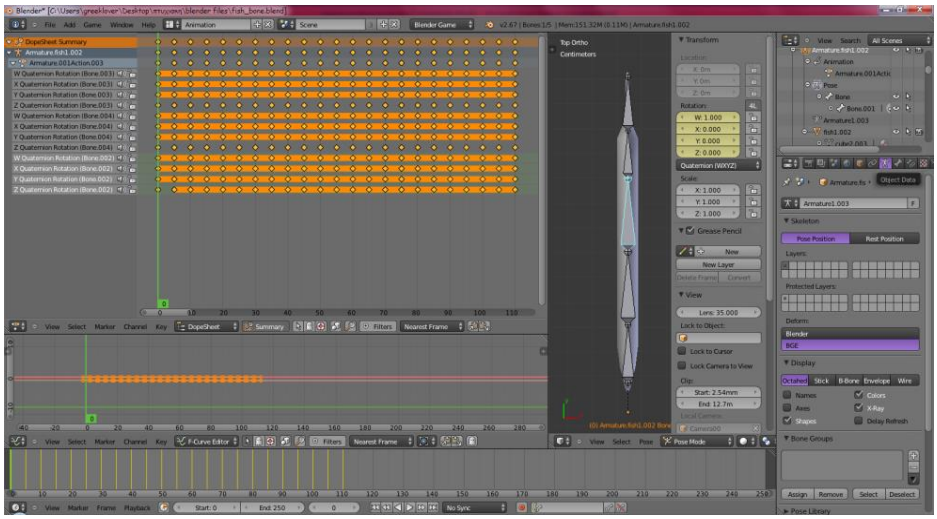
Είναι η διαδικασία με την οποία μπορούμε να δώσουμε κινητικές ικανότητες και να παραμορφώσουμε τα αντικείμενά μας.

Θα χρησιμοποιήσουμε τη μέθοδο αυτή σε αντικείμενα, τα οποία πρέπει να φαίνεται ότι κάνουν κάποια κίνηση, όπως είναι τα ψάρια και τα φύκια. Ο τύπος αντικειμένου που χρησιμοποιείται για την υλοποίηση του Rigging είναι το **Armature**, το οποίο βασίζεται σε πραγματικούς σκελετούς που συναντάμε στην πραγματικότητα. Τα βασικά συστατικά ενός Armature είναι τα **Bones**, με τα οποία σχηματίζουμε έναν σκελετό για ένα αντικείμενο [68].



Εικόνα 18 – η ιεραρχία των Bones που βάλαμε στο Mesh ψάρι

Στη συνέχεια με τη χρήση animation θα κάνουμε τα bones που φτιάξαμε να κινούνται [45].

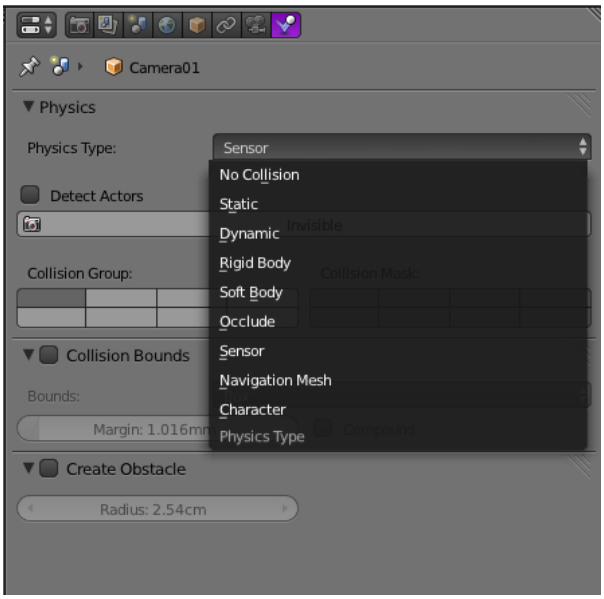


Εικόνα 19 – οι Editors για τοAnimation που θα βάλουμε στα Bones

3.3.1.11 Physics

Το Blender περιλαμβάνει προσομοίωση φυσικών ιδιοτήτων, με τη μορφή της μηχανής Bullet Physics η οποία μπορεί να χρησιμοποιηθεί για παιχνίδια, αλλά και για τα κινούμενα σχέδια. [46]

Οι επιλογές της καρτέλας των Physics είναι διαφορετικές για κάθε μηχανή του Blender.



Εικόνα 20 – Physics

Οι τύποι που περιέχει η Blender Game Engine είναι:

- No Collision: τα αντικείμενα με αυτόν τον τύπο μένουν ανεπηρέαστα από φυσικά αίτια, ενώ μπορούν να προκαλέσουν φυσικές αντιδράσεις.
- Static: δεν επηρεάζεται από φυσικά αίτια, αλλά μπορεί να επιστρέψει αντιδράσεις σύγκρουσης
- Dynamic: δέχεται και επιστρέφει αντιδράσεις. Όταν προκαλεί αντίδραση, μένει ανεπηρέαστο
- Rigid Body: δέχεται και επιστρέφει αντιδράσεις, αλλάζοντας την ταχύτητα και την περιστροφή του, ανάλογα την αντίδραση
- Soft Body: αναπαράσταση του μαλακού σώματος, για παράδειγμα ύφασμα
- Occluder: αποτρέπει τον υπολογισμό των αποδιδόμενων αντικειμένων
- Sensor: το αντικείμενο λειτουργεί σαν αισθητήρας που ανιχνεύει παρουσία άλλων αντικειμένων χωρίς επιστρεφόμενες συγκρούσεις
- Navigation Mesh: το αντικείμενο χρησιμοποιείται για την εύρεση μονοπατιών

Σε αντικείμενα όπως το Ship, Bridge, treasure_chest, stone, sword, τα οποία, είναι ακίνητα και δεν εκτελούν κάποια ενέργεια, ταιριάζει το Static. Ο λόγος που δεν χρησιμοποιούμε το No Collision, είναι επειδή παρόλο που τα objects αυτά, είναι ακίνητα, πρέπει να αλληλεπιδρούν με άλλα αντικείμενα τα οποία κινούνται. Όπως για παράδειγμα τα ψάρια τα οποία όποτε αγγίζουν αυτά τα αντικείμενα περιστρέφονται και συνεχίζουν την πορεία τους.

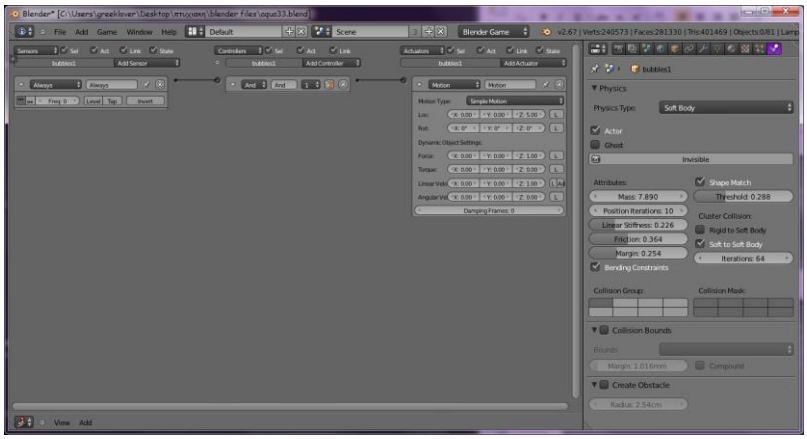
Τα ψάρια, όπως είναι τα objects Fish και Acantharus, βάλαμε το Sensor, καθώς Sensor βάλαμε και στα Armatures τους. Τα κάναμε δηλαδή να λειτουργούν σαν αισθητήρες έτσι ώστε όταν ανιχνεύουν αντικείμενα, τα οποία ορίσαμε στο Logic Editor, να εκτελούν κάποιες ενέργειες.

Το Hook (αγκίστρι-παίχτης) είναι και αυτό Static, αλλά τσεκάρουμε την επιλογή Actor, έτσι ώστε όποτε έρχεται σε επαφή με αντικείμενα τα οποία έχουν έναν Near Sensor στο Logic Editor, να το βρίσκει και να ενεργοποιείται ο αντίστοιχος Actuator του Near Sensor. Όπως και στο παιχνίδι, όπου το αγκίστρι όταν ακουμπάει ένα ψάρι, αυτό εξαφανίζεται, όπως έχουμε το έχουμε ρυθμίσει στον Logic Editor.

Το καμάκι το κάνουμε Dynamic έτσι ώστε να είναι η κίνηση του πιο ρεαλιστική σύμφωνα με τουν φυσικούς νόμους και τσεκάρουμε επίσης το Actor.

Στα seaweed και τα armatures τους βάζουμε No Collision.

Το Bubble object που βγαίνει από τα ψάρια και από κάποια άλλα αντικείμενα, το κάναμε Soft Body. Στην εικόνα 21, φαίνονται οι ρυθμίσεις του Physics, καθώς και ο Logic Editor για την κίνηση που θα κάνει. Στον Motion Actuator παρατηρούμε ότι η επιλογή Simple Motion έχει παραπάνω ρυθμίσεις.



Εικόνα 21 – Bubble Physics και Logic Editor

Οι κάμερες, οι λάμπες και τα Empty που χρησιμοποιήσαμε, έχουν από προεπιλογή το No Collision το οποίο θα αφήσουμε έτσι, καθώς κατά το τρέξιμο του παιχνιδιού, δεν θέλουμε να έχουν καμία ενέργεια και επιρροή.

3.3.1.12 Blender Game Engine

a. Εισαγωγή [47]

Το Game Engine είναι μία λειτουργία του Blender που χρησιμοποιείται για την κατασκευή διαδραστικού περιεχομένου σε πραγματικό χρόνο. Είναι ένα λογισμικό ανεξαρτήτου πλατφόρμας, το οποίο έχει γραφτεί σε γλώσσα προγραμματισμού C++ και περιλαμβάνει υποστήριξη για λειτουργίες όπως η Python scripting και OpenAL ήχο.

Τα χαρακτηριστικά της είναι:

- Χρησιμοποιεί ένα σύστημα γραφικών, τα λεγόμενα “logic bricks” για να ελέγχει την κίνηση και την απεικόνιση των αντικειμένων χωρίς τη χρήση προγραμματισμού, δίνοντας όμως τη δυνατότητα επέκτασης με τη χρήση κάποιων βιβλιοθηκών της Python.
- Ανιχνεύει συγκρούσεις και παρέχει προσωμοίωση δυναμικής με τη χρήση των Physics.
- Υποστηρίζει τύπους σχημάτων όπως κύβος, σφαίρα, κύλινδρος, κώνος, κάψουλα ένωση.
- Υποστηρίζει πλήρως τη δυναμική οχημάτων, όπως τριβή ελαστικών, δυσκαμψία, κλπ.
- Παρέχει Python scripting για προηγμένο έλεγχο τεχνητής νοημοσύνης.
- Υποστηρίζει πλήρως όλους τους τρόπους φωτισμού OpenGLTM.
- Υποστηρίζει τη χρήση πολλαπλών materials και textures, διάφορους τρόπους ανάμιξης υφών, φωτισμό ανά pixel, δυναμικό φωτισμό, τρόπους χαρτογράφησης, κινούμενα υλικά.
- Αναπαραγωγή παιχνιδιών και διαδραστικού τρισδιάστατου περιεχομένου χωρίς μεταγλώτισση ή προεπεξεργασία.
- Υποστήριξη ήχου.

OpenGLTM [48]

Η OpenGL (Open Graphics Library) είναι ένα λογισμικό ανεξαρτήτου πλατφόρμας το οποίο αναπτύχθηκε απο την Silicon Graphics και διαχειρίζεται απο τον όμιλο Khronos. Είναι μία διεπαφή προγραμματισμού εφαρμογών (API) για την απόδοση δισδιάστατων και τρισδιάστατων διανυσματικών γραφικών. Χρησιμοποιείται ευρέως σε CAD, εικονική πραγματικότητα, επιστημονικής απεικόνισης, οπτικοποίησης πληροφοριών, προσομοίωσης πτήσης, και βιντεοπαιχνίδια.

Η OpenGL Shading Language (GLSL), είναι μιά γλώσσα προγραμματισμού υψηλού επιπέδου, η οποία είναι προσαρμοσμένη να προγραμματίζει εφέ σκίασης για τρισδιάστατα γραφικά. Δημιουργήθηκε από την OpenGL και βασίζεται στη σύνταξη της γλώσσας προγραμματισμού C. [48]

b. Game Logic Editor [49]

Το πιο σημαντικό κομμάτι της μηχανής Game Engine του Blender, είναι ο Game Logic Editor. Παρέχει την κύρια μέθοδο για τη δημιουργία και την επεξεργασία της λογικής του παιχνιδιού για όλα τα αντικείμενα που συνθέτουν το παιχνίδι.

Χωρίζεται σε δύο μέρη, το πεδίο Properties και το πεδίο με τα Logic Bricks.

Στο Properties προσθέτουμε στο παιχνίδι ιδιότητες όπως παράδειγμα το μέτρημα των πόντων και τη διάρκεια ζωής του παίχτη. [50]

Στο δεύτερο πεδίο έχουμε τις τρεις κατηγορίες των Logic Bricks· Sensors, Controllers, Actuators.

Απο το πεδίο Sensors διαλέγουμε με ποιό τρόπο θα δίνουμε εντολή για να γίνεται μία ενέργεια. Ανοίγοντας τη λίστα, βλέπουμε ότι έχει διάφορους τύπους (Εικόνα 89). [51]

- Actuator: ανιχνεύει πότε κάποιος actuator θα λάβει έναν παλμό ενεργοποίησης
- Always: δίνει συνεχώς ένα σήμα εξόδου
- Collision: ανιχνεύει συγκρούσεις μεταξύ αντικειμένων ή materials
- Delay: καθυστερήση εξόδου από ένα συγκεκριμένο αριθμό λογικών παλμών.
- Joystick: ανιχνεύει είσοδο απο την κίνηση χειριστηρίου
- Keyboard: ανιχνεύει είσοδο από το πληκτρολόγιο
- Message: εντοπίζει μηνύματα ή ιδιότητες
- Mouse: εντοπίζει κινήσεις ποντικιού ως είσοδο
- Near: ανιχνεύει αντικείμενα που βρίσκονται σε κάποια απόσταση
- Property: εντοπίζει αλλαγές στις ιδιότητες του αντικειμένου ως είσοδο
- Radar: εντοπίζει αντικείμενα που κινούνται σε συγκεκριμένη απόσταση από κάποια γωνία των αξόνων
- Random: παράγει τυχαίους παλμούς
- Ray: ανιχνεύει χτυπήματα εντοπίζοντας μια ακτίνα στην κατεύθυνση του άξονα
- Touch: εντοπίζει πότε δύο αντικείμενα έρχονται σε επαφή μεταξύ τους

Το πεδίο Controllers είναι οι ελεγκτές που συλλέγουν τα εισερχόμενα δεδομένα των Sensors και προσδιορίζουν την κατάσταση. Αφού εκτελέσουν κάποιες ειδικές λειτουργίες, στέλνουν παλμούς στους Actuators με τους οποίους συνδέονται. [52]

- AND
- OR
- XOR
- NAND
- NOR
- XNOR
- Expression
- Python

Ο παρακάτω πίνακας δείχνει τα αποτελέσματα των λογικών πράξεων από τους έξι πρώτους Controllers. Στη στήλη με τους θετικούς παλμούς, έχουμε τους παλμούς που στέλνονται απο τους Sensors με τους οποίους είναι συνδεδεμένοι. Στις στήλες με τους Controllers έχουμε την ανταπόκρισή τους στους παλμούς. Το True σημαίνει ότι οι Actuator που είναι συνδεδεμένοι στους Controllers θα ενεργοποιηθούν, εκτελώντας την εντολή που δώθηκε, ενώ το False σημαίνει ότι οι Actuators δεν θα ενεργοποιηθούν και άρα καμμία εντολή δεν θα εκτελεστεί.

Θετικοί παλμοί	**Controllers**					
	AND	OR	XOR	NAND	NOR	XNOR
κανένας	False	False	False	True	True	True
ένας	False	True	True	True	False	False
πολλοί	False	True	False	True	False	True
όλοι	True	True	False	False	False	True

Controllers

Ο Controller Expression, είναι μία γραπτή έκφραση, η οποία αν είναι αληθής δίνει θετική έξοδο. Η έκφραση αυτή μπορεί να είναι σταθερές, μεταβλητές, μαθηματικές πράξεις, λογικές πράξεις ή υποθετικές εκφράσεις. [53]

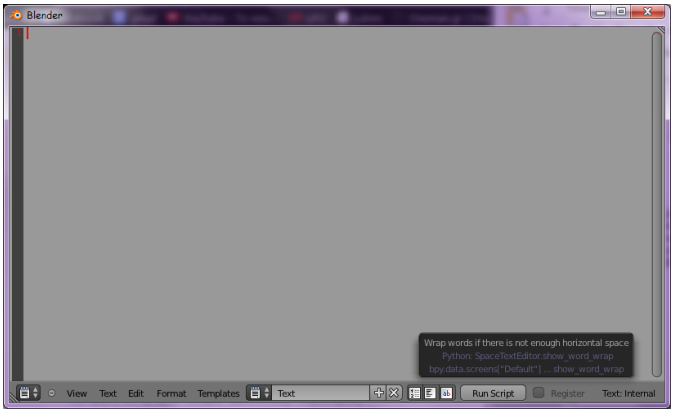
Με τον Python Controller γίνεται προσθήκη Python Scripts στο αντικείμενο. Τα Scripts γράφονται στον Text Editor.

Στο πεδίο Actuators, γίνεται η εκτέλεση των ενεργειών όπως η κίνηση, η δημιουργία αντικειμένων, η αναπαραγωγή ήχου, κλπ. Ενεργοποιούνται όταν λάβουν από τους Controllers τουλάχιστον έναν θετικό παλμό. [54]

- Action: χειρίζεται τις ενέργειες οπλισμού (armature).
- Camera: χρησιμοποιείται από κάμερες ή άλλα αντικείμενα, έτσι ώστε να ακολουθούν με ομαλότητα άλλα αντικείμενα
- Constraint: χρησιμοποιείται για να περιορίζει τον χώρο ενός αντικειμένου, ή την περιστροφή του, επίσης χρησιμεύει για τον έλεγχο των φυσικών καταστάσεων του αντικειμένου στο παιχνίδι
- Edit Object: επεξεργάζεται και αλλάζει το αντικείμενο, προσθέτει ή καταστρέφει αντικείμενα
- Filter 2D: προσθέτει φίλτρα για ειδικά εφέ, όπως θάμπωμα
- Game: απο εδώ χειριζόμαστε το παιχνίδι, επανεκκίνηση, έξοδος, φόρτωμα παιχνιδιού και αποθήκευση
- Message: στέλνει μηνύματα τα οποία ενεργοποιούν άλλα αντικείμενα
- Motion: δίνει στα αντικείμενα κίνηση ανάλογα με τις επιλογές Physics που έχουμε ορίσει
- Parent: καθορίζει την πατρότητα των αντικειμένων
- Property: χειρίζεται τις ιδιότητες των αντικειμένων
- Random: δημιουργεί τυχαίες τιμές οι οποίες χρησιμοποιούνται απο τις ιδιότητες
- Scene: διαχειρίζεται τις σκηνές στο αρχείο μας
- Sound: αναπαράγει ήχους στο παιχνίδι
- State: αλλάζει την κατάσταση του αντικειμένου
- Steering: παρέχει επιλογές εύρεσης μονοπατιού για το αντικείμενο
- Visibility: αλλάζει τον τρόπο που φαίνεται το αντικείμενο

c. Text Editor [55]

Ο Text Editor είναι η περιοχή στο Blender όπου γράφουμε τα Python Scripts. Δημιουργούμε ένα καινούριο text από το button New. Όπως φαίνεται στην εικόνα το όνομα του αρχείου είναι Text, το οποίο μπορούμε να το αλλάξουμε. Τα 3 κουμπιά δίπλα από το όνομα είναι επιλογές κειμένου, με τις οποίες αριθμείται κάθε σειρά του κειμένου, συμπυκνώνει τις γραμμές έτσι ώστε να τις βλέπουμε όλες ανεξάρτητα με το μέγεθος του Editor και χρωματίζει τις λέξεις του κώδικα που έχουν κάποια σημασία.

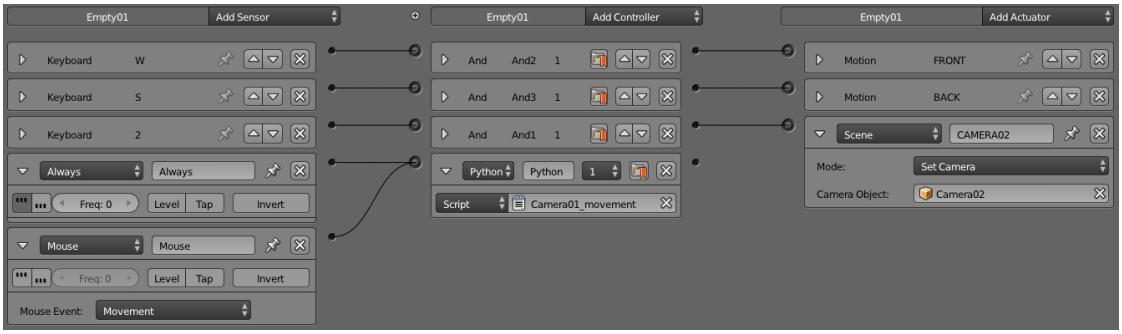


Εικόνα 22 – Text Editor

Το κουμπί Run Script τρέχει το τμήμα του κώδικα που έχουμε γράψει. Το εικονίδιο του text δίπλα στο όνομα περιέχει μία λίστα με όλα τα text files που έχουμε γράψει στο παρών .blend αρχείο. Απο το μενού, το View έχει επιλογές εμφάνισης του παραθύρου, από το Text μπορούμε να τρέξουμε το script, να το αποθηκεύσουμε. Απο το Edit κάνουμε αντιγραφή/επικόλληση του κειμένου, από το Format έχουμε επιλογές του κειμένου, όπως προσθήκη σχολίων και γράψιμο κειμένου πιο μέσα. Στο Templates έχει διάφορες έτοιμες πλατφόρμες οι οποίες περιέχουν έτοιμα script σε Python, τα οποία μπορούμε να ανοίξουμε και να τα επεξεργαστούμε ανάλογα με το τι θέλουμε να κάνουμε.

Παράδειγμα

Στην παρακάτω εικόνα βλέπουμε τα Logic Bricks για την Camera01. Οι δύο πρώτες εντολές είναι για την κίνηση μπροστά και πίσω αντίστοιχα με τη χρήση των πλήκτρων W και S. Η επόμενη είναι για την εναλλαγή μεταξύ των καμερών του παιχνιδιού, με το πλήκτρο 2 για να μεταφερθούμε στην κάμερα 2. Οι τελευταίες εντολές περιλαμβάνουν δύο Sensors τους Always και Mouse, οι οποίοι συνδέονται με έναν Python Actuator, λέγοντας έτσι , ότι με τη χρήση του ποντικιού για συνεχές διάστημα θα εκτελείται το python script που έχουμε ενώσει.



Εικόνα 23 – Logic Editor για τις ρυθμίσεις της Camera01

Στο παράρτημα [75] φαίνονται όλοι οι Logic Editors που χρησιμοποιούνται στο παιχνίδι καθώς και τα script για την κίνηση του ποντικιού και των αντικειμένων.

4. ΑΠΟΤΕΛΕΣΜΑΤΑ

4.1 Απόδοση παιχνιδιού

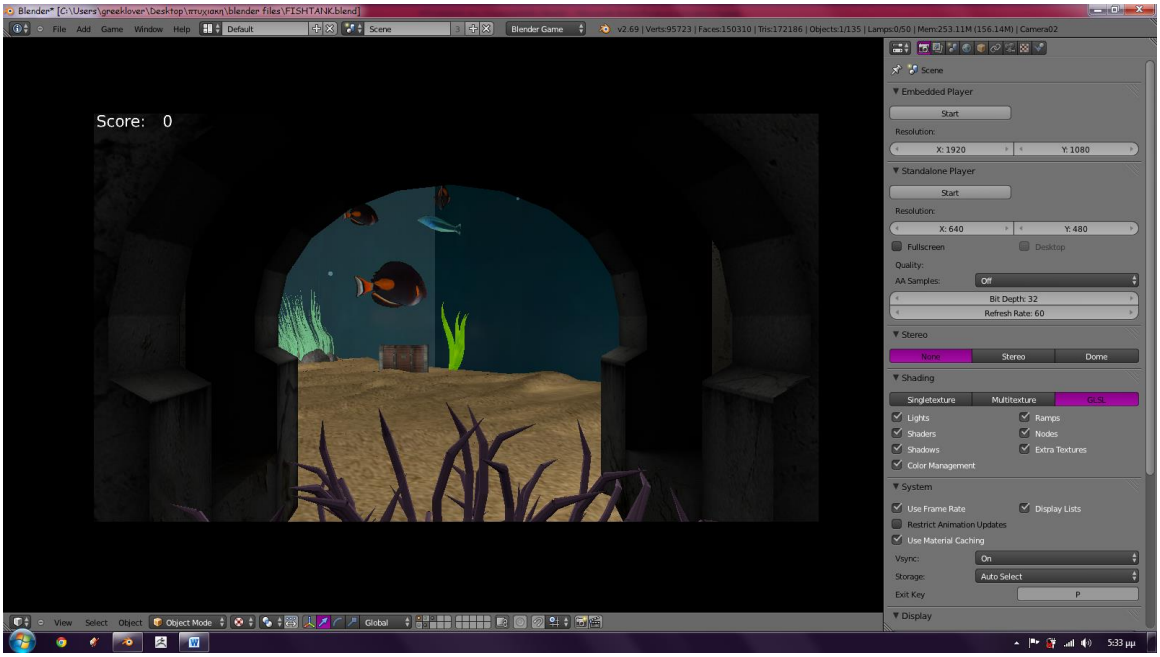
Το παιχνίδι μπορούμε να το τρέξουμε μέσω του Embedded Player και του Standalone Player[56].

4.1.1 Embedded Player

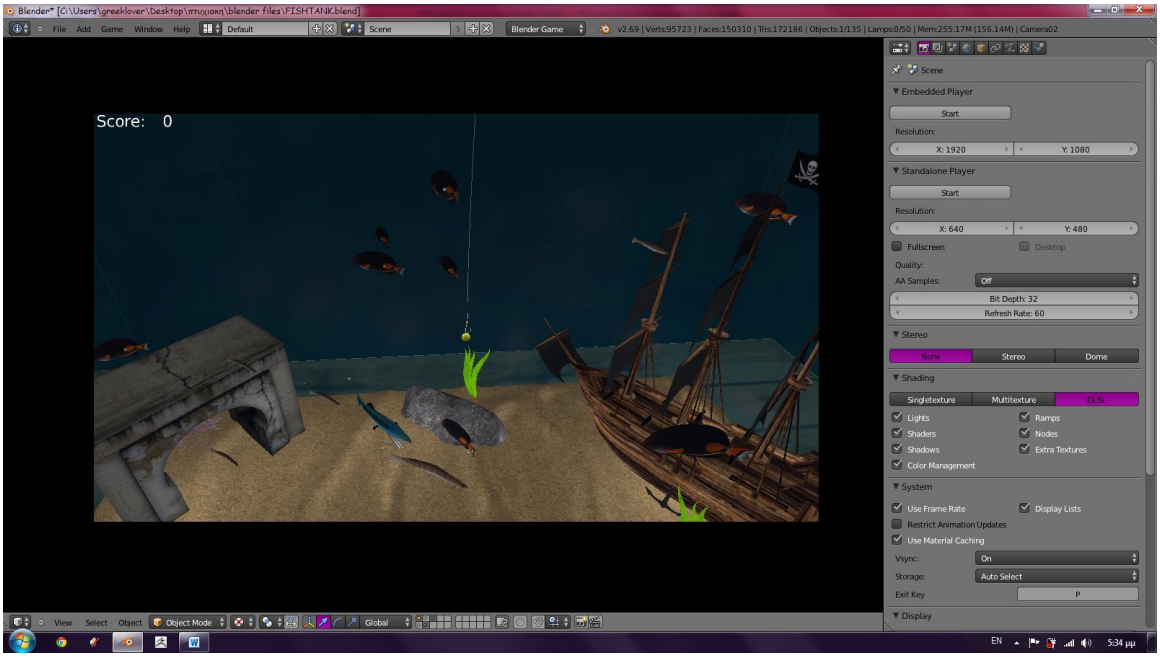
Λειτουργεί μέσα από το Blender, κάτι το οποίο μας βοηθάει κατά τη διάρκεια της δημιουργίας του παιχνιδιού, να παρακολουθούμε την πρόοδό του.

Υπάρχουν τρείς τρόποι για να τρέξουμε το παιχνίδι μέσα από την εφαρμογή. Ο πρώτος είναι από το μενού Game→Start Game Engine. Ο δεύτερος είναι από το Properties Editor στην καρτέλα Render, πατώντας το κουμπί Start του Embedded Player. Ο τρίτος είναι πατώντας το κουμπί P στον 3D Editor. Και με τους 3 τρόπους η ανάλυση του παιχνιδιού θα είναι οι τιμές στο Embedded Player.

Το αποτέλεσμα θα είναι κοινό επίσης και για τους 3 τρόπους, οι παρακάτω εικόνες δείχνουν ένα στιγμιότυπο για κάθε κάμερα μέσω του Embedded Player.



Εικόνα 24 - Όψη Camera01 απο τον Embedded Player



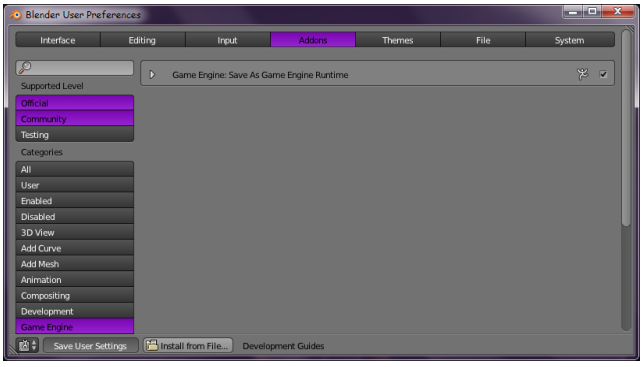
Εικόνα 25 - Όψη Camera02 απο τον Embedded Player

4.1.2 Standalone Player [56]

Το παιχνίδι τρέχει σαν εκτελέσιμο αρχείο, χωρίς να είναι απαραίτητο να ανοίξουμε το Blender ή να το έχουμε στον υπολογιστή μας. Απο την επιλογή Standalone Player στο Properties Editor, πατώντας το Start ανοίγουμε ένα νέο παράθυρο στο οποίο ξεκινά να τρέχει το παιχνίδι. Στο Resolution από κάτω επιλέγουμε την ανάλυση του παραθύρου, πατώντας το Fullscreen κάνουμε το παιχνίδι να εμφανίζεται σε όλη την οθόνη, ενώ το Desktop το περιορίζει στο μέγεθος της επιφάνειας εργασίας του υπολογιστή.

Για να κάνουμε το παιχνίδι να μπορεί να τρέξει χωρίς να είναι απαραίτητα ανοιχτό το Blender, ακολουθούμε την εξής διαδικασία:

- Απο το μενού File ανοίγουμε το User Preferences [57]. Στην καρτέλα Addons διαλέγουμε την κατηγορία Game Engine και τσεκάρουμε το Save As Game Engine Runtime. Δίνουμε έτσι την δυνατότητα στο Blender να μπορέσει να εξάγει το παιχνίδι σαν ένα εκτελέσιμο αρχείο.
- Πατάμε File→Extenal Data→Pack into .blend το οποίο πακετάρει όλα τα εξωτερικά αρχεία που έχουμε χρησιμοποιήσει όπως τις εικόνες για τα textures στο αρχείο .blend.
- Ρυθμίζουμε από την καρτέλα Standalone Player την ανάλυση, και σιγουρευόμαστε ότι έχουμε επιλέξει την σωστή κάμερα, απο την οποία βλέπουμε.
- Πάμε απο το File→Export→Save As Game Engine Runtime. Προσέχουμε να είναι επιλεγμένα κάτω αριστερά το Copy Python Copy DLLs και δημιουργούμε έναν νέο φάκελο από το Create New Directory. μέσα στο φάκελο αυτό πατάμε το κουμπί Save As Game Engine Runtime.

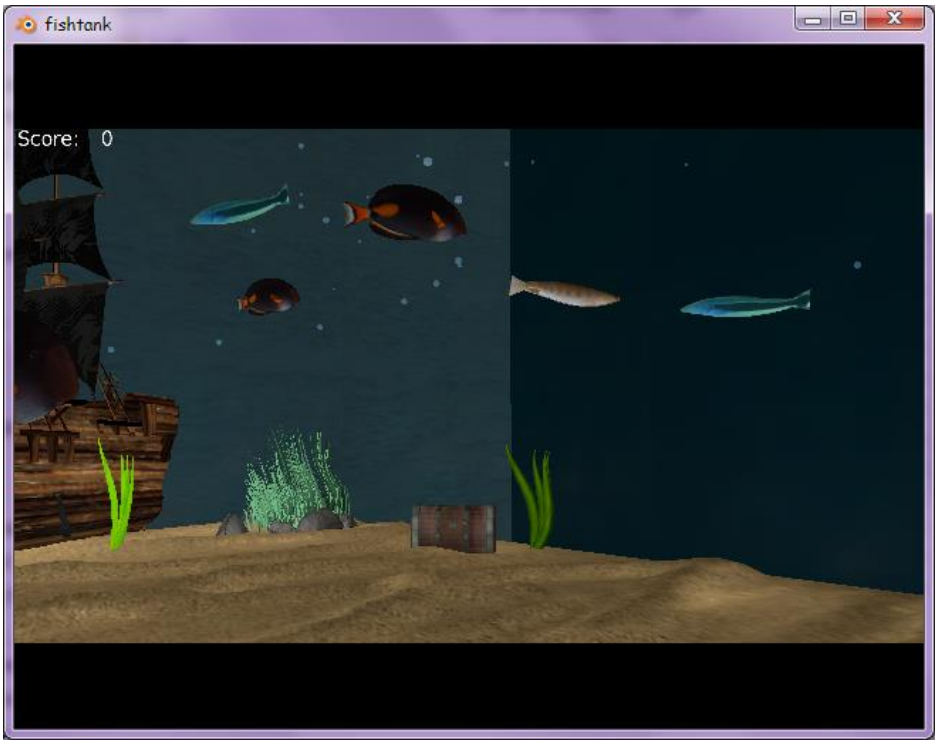


Εικόνα 26 - Ενεργοποίηση του Save As Game Engine Runtime

Πρέπει να προσέξουμε αν στο φάκελο που εξάγαμε το παιχνίδι με την κατάληξη .exe, υπάρχουν και τα .dlls του καθώς και ένας φάκελος με τις βιβλιοθήκες της Python.

Επίσης, πρέπει ο φάκελος του εκτελέσιμου να είναι μαζί με τους φακέλους που έχουμε τα texture και ότι άλλο χρησιμοποιήσαμε για την υλοποίηση του παιχνιδιού, έτσι ώστε να μπορούμε να τρέχουμε με επιτυχία το παιχνίδι.

Στις παρακάτω εικόνες βλέπουμε το παράθυρο του εκτελέσιμου αρχείου του παιχνιδιού με την κατάληξη .exe. απο τις Camera01 και Camera02.



Εικόνα 27 - Standalone Player Camera01

Εικόνα 28 - Standalone Player Camera02

4.2 Συμπεράσματα

Στην αρχή της υλοποιήσης της πτυχιακής εργασίας, αποδείχτηκε ότι το SketchUp δεν ήταν το κατάλληλο πρόγραμμα για τους σκοπούς της εργασίας. Είναι μία εφαρμογή καθαρά για τον σχεδιασμό τρισδιάστατων αρχιτεκτονικών σχεδίων. Έτσι ο σχεδιασμός συνεχίστηκε στο σχεδιαστικό πρόγραμμα Blender.

Ο μόνος τρόπος για να «επικοινωνήσουν» τα δύο προγράμματα μεταξύ τους, ήταν ο τύπος αρχείων Collada. Έτσι κάναμε export το μοντέλο απο το SketchUp σε μορφή Collada και το κάναμε import στο Blender με την ίδια μορφή. Κατα τη διαδικασία εισαγωγής/εξαγωγής μεταξύ των δύο προγραμμάτων, έγιναν δοκιμές για να διαπιστώσουμε με ποιόν τρόπο θα μεταφέραμε το μοντέλο στο Blender ώστε να είναι όσο το δυνατόν πιο εύχρηστο στο καινούριο πρόγραμμα, αφού οι πλευρές των αντικειμένων του μοντέλου, διαιρούνταν με την εισαγωγή στο Blender, κάνοντας έτσι πιο δύσκολη την επεξεργασία τους.

Ανοίγοντας το Blender είδαμε ένα πολύπλοκο interface, είναι σχεδιασμένο με διαφορετικό τρόπο απο άλλα παρόμοια προγράμματα, προσπαθώντας να προσφέρει ένα περιβάλλον πιο φιλικό προς το χρήστη.

Τα πλεονεκτήματά που διαπιστώσαμε είναι:

- Είναι ελεύθερο λογισμικό, έχει μικρό μέγεθος και έιναι διαθέσιμο σε Windows, Linux, Mac OS
- Οι περισσότερες ενέργειές του γίνονται με τη χρήση συντομεύσεων πληκτρολογίου
- Δίνει τη δυνατότητα στο χρήστη να προσαρμόσει το interface της εφαρμογής όπως τον βολεύει και του αρέσει
- Διαθέτει το Outliner το οποίο δείχνει όλες τις σκηνές και τα αντικείμενα που βρίσκονται σε αυτές ιεραρχημένα
- Εύκολη χρήση και στην τοποθέτηση Armature πάνω σε ένα αντικείμενο, με την επιπλέον χρήση των εργαλείων για Animation που διαθέτει, ώστε ο σκελετός του επισυνάπτουμε σε ένα αντικείμενο να έχει και τη δυνατότητα κίνησης
- Περιέχει αναλυτικές ρυθμίσεις για τη δημιουργία Animation
- Για τη δημιουργία παιχνιδιών διαθέτει τα Logic Bricks έναν γραφικό τρόπο για να δίνουμε εντολές στα αντικείμενα και επίσης διαθέτει ενσωματωμένο επεξεργαστή κειμένου για Python scripting
- Είναι συμβατό με πολλούς τύπους αρχείων όπως Collada, 3D Studio, Wavefront, Autodesk FBX

Οι τομείς στους οποίους είδαμε ότι υστερεί είναι:

- Είναι δύσκολο και χρονοβόρο στην εκμάθησή του ώστε να μπορέσει ο χρήστης να το χρησιμοποιήσει παραγωγικά
- Στον γραφικό σχεδιασμό αντικειμένων· παρόλο που έχει έτοιμα τρισδιάστατα σχήματα, είναι δύσκολο να σχεδιάσεις ένα πολύπλοκο μοντέλο όπως για παράδειγμα ένα ανθρώπινο σώμα ή κεφάλι
- Περίπλοκη χρήση των Materials και Textures και η χρήση των UV Maps
- Rigid ή Soft boddies που τυχόν έχουμε βάλει για physics σε αντικείμενα, καθυστερούν πολύ τη φόρτωση και την απόδοση του παιχνιδιού
- Αλλάζουν βασικές ρυθμίσεις όπως των Materials, ανάμεσα στις τρεις μηχανές που διαθέτει το Blender

Σε γενικές γραμμές η χρήση των δύο προγραμμάτων ήταν ικανοποιητική. Υπηρχε δυσκολία για την εκμάθησή και την κατανόησή τους, αλλά αυτό το εμπόδιο ξεπεράστικε σχετικά γρήγορα. Βρέθηκαν οι δυνατότητες και οι αδυναμίες τους, και η χρήση για την οποία είναι κατάλληλα.
Το τελικό αποτέλεσμα ήταν επίσης ικανοποιητικό, αν και πολλές φορές ήταν δύσκολη η διαδικασία ολοκλήρωσής του.

5. ΒΙΒΛΙΟΓΡΑΦΙΑ

http://en.wikipedia.org/wiki/Video_game
http://en.wikipedia.org/wiki/Video_game_genres
http://en.wikipedia.org/wiki/First-person_shooter
http://en.wikipedia.org/wiki/Massively_multiplayer_online_first_person_shooter
http://en.wikipedia.org/wiki/Massively_multiplayer_online_role-playing_game
http://en.wikipedia.org/wiki/Doom_(video_game)
http://en.wikipedia.org/wiki/Call_of_Duty_(video_game)
http://en.wikipedia.org/wiki/Resident_Evil
http://en.wikipedia.org/wiki/World_of_Warcraft
http://en.wikipedia.org/wiki/Warcraft
http://en.wikipedia.org/wiki/The_Sims
http://en.wikipedia.org/wiki/Pro_Evolution_Soccer
http://en.wikipedia.org/wiki/FIFA_(video_game_series)
http://en.wikipedia.org/wiki/3D_engine
http://en.wikipedia.org/wiki/Unreal_Engine
http://www.unrealengine.com/en/
http://en.wikipedia.org/wiki/Doom_engine
http://en.wikipedia.org/wiki/Anvil_(game_engine)
http://en.wikipedia.org/wiki/Crystal_Tools
http://en.wikipedia.org/wiki/Panda3D
http://en.wikipedia.org/wiki/Autodesk_Maya
http://en.wikipedia.org/wiki/Unity_Web_Player
http://unity3d.com/
http://en.wikipedia.org/wiki/JMonkeyEngine
http://jmonkeyengine.org/
http://en.wikipedia.org/wiki/IW_Engine
http://en.wikipedia.org/wiki/Game_Blender
http://en.wikipedia.org/wiki/CryEngine
http://en.wikipedia.org/wiki/Autodesk_3ds_Max
http://en.wikipedia.org/wiki/SketchUp#Google_SketchUp
http://www.sketchup.com/
http://help.sketchup.com/en/article/73815
http://sketchup.google.com/3dwarehouse/details?mid=3469cdf83a3cba1f35836c728d324152&prevstart=48
http://help.sketchup.com/en/article/151692
http://www.google.gr/books?hl=en&lr=&id=CHKdorM71c8C&oi=fnd&pg=PT2&dq=exporting+from+google+sketchup&ots=snFZ#v=onepage&q=exporting%20from%20google%20sketchup&f=false
http://www.google.gr/books?hl=en&lr=&id=Vrz60NEqw0gC&oi=fnd&pg=PR5&dq=exporting+from+google+sketchup&ots=UWDl#v=onepage&q=exporting%20from%20google%20sketchup&f=false
http://www.google.gr/books?hl=en&lr=&id=1k3uczGGC3oC&oi=fnd&pg=PA1&dq=exporting+from+google+sketchup&ots=WmSe#v=onepage&q=exporting%20from%20google%20sketchup&f=false
http://www.google.gr/books?hl=en&lr=&id=NB1ezAb-MDwC&oi=fnd&pg=PA4&dq=exporting+with+google+sketchup&ots=fLtvzXYHHB&sig=ckg_vkNEwK_TZ45NQ47dk5C08nw&redir_esc=y#v=onepage&q=exporting%20with%20google%20sketchup&f=false
http://support.google.com/sketchup/bin/answer.py?hl=en&answer=140409
http://www.dummies.com/how-to/content/how-to-export-an-animation-with-google-sketchup-8.html
http://en.wikipedia.org/wiki/COLLADA

Streamlining workflow in architectural visualization with Blender and SketchUp - Jani Lintunen
Sound in COLLADA - Shih-Han Chan, Cecile Le Prado, Stéphane Natkin, Guillaume Tiger, and Alexandre Topol
http://www.khronos.org/
https://collada.org/mediawiki/index.php/COLLADA
http://en.wikipedia.org/wiki/Blender_(software)
http://www.blender.org/features/
http://vr.arch.uth.gr/VR-Arch/05_Basic_Design_Tools/blender.html
http://opensci.grnet.gr/os_catalog/softwares/blender
http://library.kiwix.org/wikipedia_el_all/A/Blender.html
http://el.wikipedia.org/wiki/Blender
http://en.wikipedia.org/wiki/Blender_(software)
http://wiki.blender.org/index.php/Doc:EL/2.6/Manual/Game_Engine
http://wiki.blender.org/index.php/Doc:EL/2.6/Manual/3D_interaction/Navigating
http://en.wikipedia.org/wiki/Wavefront_.obj_file
http://en.wikipedia.org/wiki/.3ds
http://www.turbosquid.com/
http://wiki.blender.org/index.php/Doc:2.6/Manual/Introduction
http://wiki.blender.org/index.php/Doc:2.6/Manual/Game_Engine/Camera
http://wiki.blender.org/index.php/Doc:2.6/Manual/Lighting/Lamps
http://www.youtube.com/watch?v=3x4wi-clnPs
http://wiki.blender.org/index.php/Doc:2.4/Books/GameKit_2/13.Reference
http://en.wikipedia.org/wiki/B%C3%A9zier_curve
http://wiki.blender.org/index.php/Doc:2.4/Tutorials/Modeling/Curves/B%C3%A9zier
http://wiki.blender.org/index.php/Doc:2.6/Manual/Modeling/Texts
http://www.youtube.com/watch?v=rtbpv6p7jjk
http://wiki.blender.org/index.php/Doc:EL/2.6/Manual/Materials/Assigning_a_material
http://wiki.blender.org/index.php/Doc:EL/2.6/Manual/Textures/Options/Mapping
http://wiki.blender.org/index.php/Doc:EL/2.6/Manual/Textures/Options/Texture_Panel
http://www.luxrender.net/wiki/LuxRender_Textures_Blender
http://www.tutorialsforblender3d.com/Textures/Textures_index.html
http://www.cgtextures.com/index.php
http://wiki.blender.org/index.php/Doc:2.4/Manual/Textures
http://wiki.blender.org/index.php/Doc:2.4/Manual/Textures/Mapping/UV
http://wiki.blender.org/index.php/Doc:2.4/Manual/Textures/Mapping/UV/Unwrapping
http://wiki.blender.org/index.php/Doc:2.4/Manual/Textures/Mapping/UV/Layout_Management
http://wiki.blender.org/index.php/Doc:2.4/Manual/Textures/Mapping/UV/Layout_Editing
http://wiki.blender.org/index.php/Doc:2.4/Manual/Textures/Mapping/UV/Applying_Image
http://wiki.blender.org/index.php/Doc:2.4/Manual/Animation/Editors
http://wiki.blender.org/index.php/Doc:2.6/Manual/Rigging/Armatures
http://wiki.blender.org/index.php/Doc:2.4/Manual/Animation
http://wiki.blender.org/index.php/Doc:2.6/Manual/Rigging
http://en.wikibooks.org/wiki/Blender_3D:_Noob_to_Pro/Bones
http://wiki.blender.org/index.php/Doc:2.4/Manual/Rigging/Skinning/Objects
http://wiki.blender.org/index.php/Doc:2.6/Manual/Game_Engine/Logic/Editor
http://en.wikipedia.org/wiki/OpenGL_Shading_Language
http://en.wikipedia.org/wiki/Shading_language#OpenGL_shading_language
http://wiki.blender.org/index.php/Doc:2.6/Manual/Game_Engine/Logic/Sensors
http://wiki.blender.org/index.php/Doc:2.6/Manual/Game_Engine/Logic/Controllers

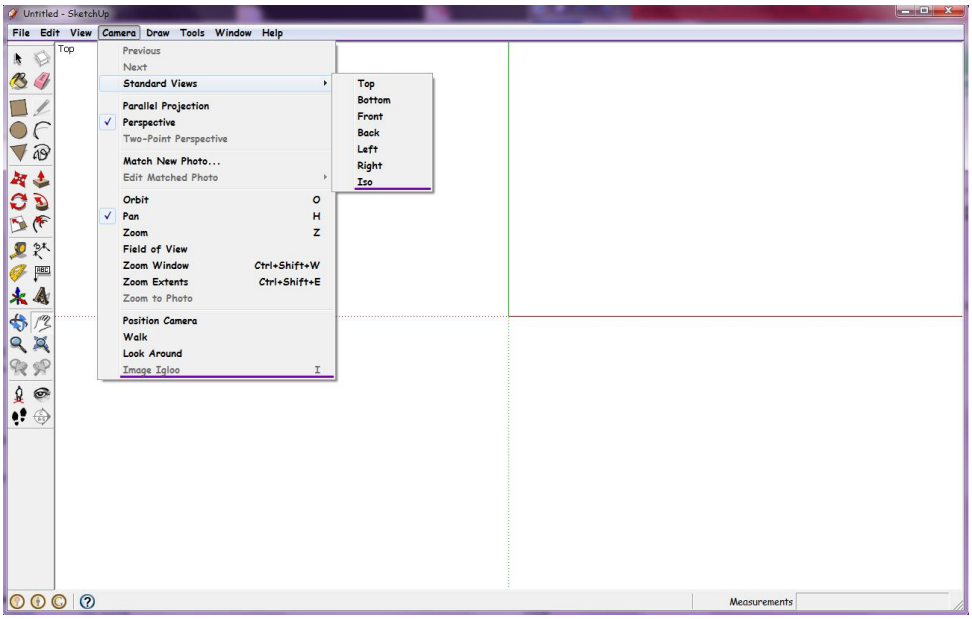
http://wiki.blender.org/index.php/Doc:2.6/Manual/Game_Engine/Logic/Actuators
http://wiki.blender.org/index.php/Doc:2.6/Manual/Extensions/Python/Text_editor
http://wiki.blender.org/index.php/Doc:2.6/Manual/Modeling/Empties
http://www.blender.org/documentation/blender_python_api_2_69_1/bge.logic.html
http://www.blender.org/documentation/blender_python_api_2_69_1/bge.render.html
http://www.blender.org/documentation/blender_python_api_2_69_1/bge.types.KX_Scene.html
http://www.blender.org/documentation/blender_python_api_2_69_1/mathutils.html
http://www.blender.org/documentation/blender_python_api_2_69_1/bge.types.KX_GameObject.html
http://wiki.blender.org/index.php/Doc:2.6/Manual/Game_Engine/Physics
http://wiki.blender.org/index.php/User:Sculptorjim/Game_Engine/Physics/Objects/No_Collision
http://wiki.blender.org/index.php/User:Sculptorjim/Game_Engine/Physics/Objects/Static
http://wiki.blender.org/index.php/User:Sculptorjim/Game_Engine/Physics/Objects/Dynamic
http://wiki.blender.org/index.php/User:Sculptorjim/Game_Engine/Physics/Objects/Rigid_Body
http://wiki.blender.org/index.php/User:Sculptorjim/Game_Engine/Physics/Objects/Soft_Body
http://wiki.blender.org/index.php/Doc:2.6/Manual/Modifiers
http://wiki.blender.org/index.php/Doc:2.6/FAQ/Game_Engine/Standalone_game#Blender_only
http://wiki.blender.org/index.php/Doc:2.6/Manual/Game_Engine/Blender_Player
http://wiki.blender.org/index.php/Doc:2.6/Manual/Preferences
http://www.blender.org/documentation/blender_python_api_2_63_20/contents.html

ΠΑΡΑΡΤΗΜΑ

A. SketchUp

A.1 Όψη κάμερας και βασικά εργαλεία

Απο το μενού ανοίγουμε ένα νέο αρχείο, έτσι ώστε να μας το ανοίξει με το πρότυπο που επιλέξαμε. Η προεπιλεγμένη όψη είναι η Top View, για να την αλλάξουμε πάμε στο μενού **Camera→Standard Views** και επιλέγουμε την όψη που θέλουμε.



Εικόνα 29 – interface του SketchUp και οι επιλογές όψης του

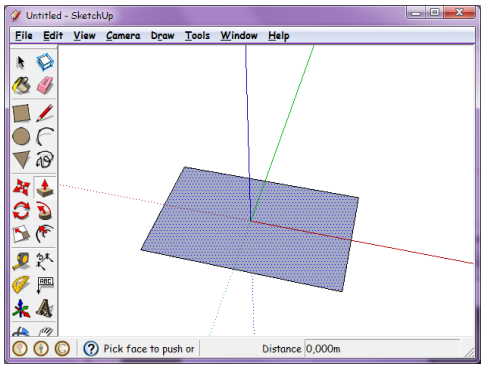
Κάθε φορά που το κάνουμε αυτό, παρατηρούμε ότι αλλάζουν και οι άξονες που φαίνονται στην οθόνη. Ο κόκκινος άξονας συμβολίζει τον άξονα των x, ο πράσινος τον άξονα των y και ο μπλε των z. Στα αριστερά της οθόνης έχουμε το παράθυρο με τα εργαλεία που χρησιμοποιούνται για το σχεδιασμό των αντικειμένων, τα οποία μπορούμε να βρούμε από το μενού **View→Toolbars→Large Tool Set**.

A.1.1 Σχεδιασμός του βασικού σχεδίου

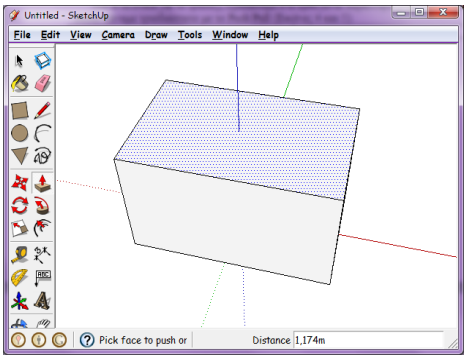
Τα αντικείμενα που φτιάξαμε με το SketchUp είναι το τραπέζι, το ενυδρείο και η βάση του, η λάμπα και το δωμάτιο όπου υπάρχουν όλα αυτά.

Πιό αναλυτικά, για το τραπέζι, σχεδιάσαμε ένα ορθογώνιο παραλληλόγραμμο με το Rectangle το οποίο κάναμε τρισδιάστατο με το Push/Pull. Στη συνέχεια μετρήσαμε με το Dimension και το Tape Measure Tool, τα σημεία στα οποία θέλουμε να δημιουργήσουμε τα κοψίματα για τη δημιουργία των ποδιών του τραπεζιού. Έπειτα φτιάχνουμε ένα δεύτερο ορθογώνιο παραλληλόγραμμο, πάνω στα σημεία που θέλουμε να έχουμε κενά, και με το Push/Pull δημιουργούμε μία τρύπα η οποία να διαπερνάει το αρχικό κουτί. Αφού φτιάξουμε με αυτόν το τρόπο το σχήμα του τραπεζιού με το Line Tool σχεδιάζουμε μία γραμμή σε κάθε πλευρά του τραπεζιού. Τέλος επιλέγουμε την πάνω επιφάνεια και με το Scale Tool την διαμορφώνουμε. Οι πάνω ενέργειες φαίνονται στις εικόνες 30 έως 34.

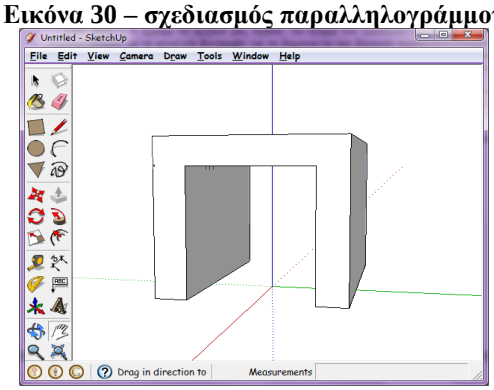
Για τη γυάλα φτιάξαμε ένα κουτί, όπως κάναμε πριν για το τραπέζι, και επιλέγοντας την πάνω επιφάνειά του, πατήσαμε delete. Για τη βάση φτιάξαμε ένα απλό κουτί, ενώ για το δωμάτιο προσθέσαμε 5 δισδιάστατα ορθογώνια παραλληλόγραμμα, ένα για τους 3 τοίχους, ένα για το ταβάνι και ένα για το πάτωμα.



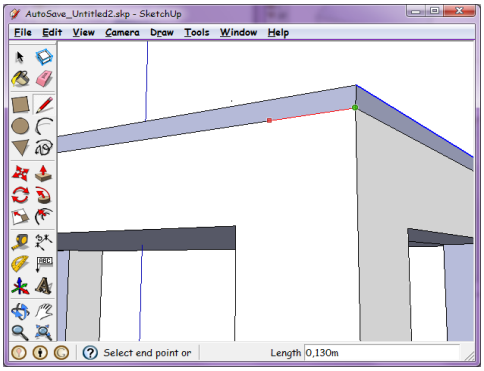
Εικόνα 30 – σχεδιασμός παραλληλογράμμου



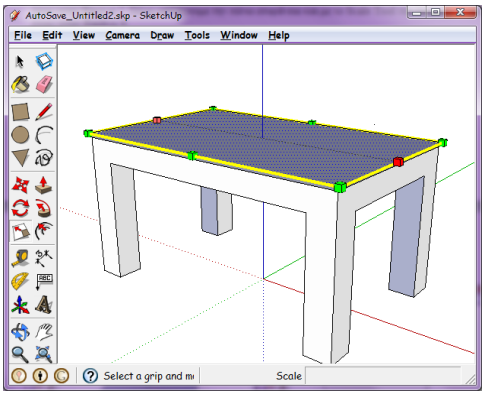
Εικόνα 31 – επέκταση του, σε 3D σχέδιο



Εικόνα 32 – αρχικοί σχέδιο τραπεζιού



Εικόνα 33 – προσθήκη έξτρα γραμμών



Εικόνα 34 – μεγάλωμα επιφάνειας

A.2 Materials – Styles

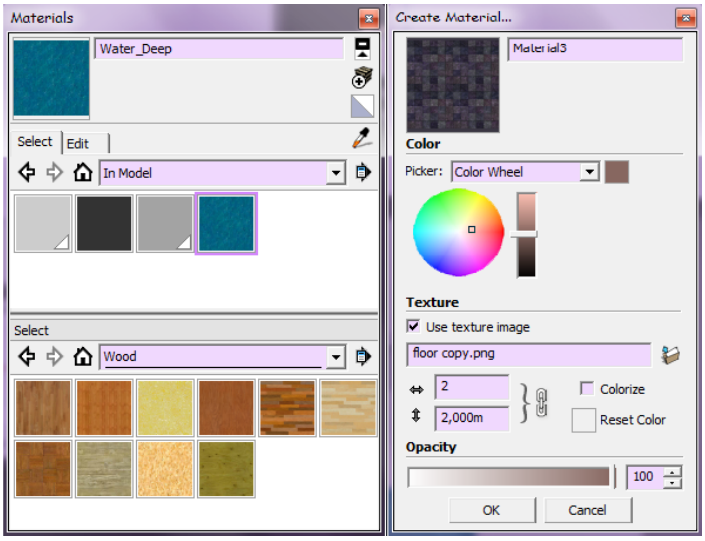
Το SketchUp περιέχει την προσθήκη υφών και μας επιτρέπει να ορίσουμε διάφορα στυλ στα αντικείμενα.

A.2.1 Materials

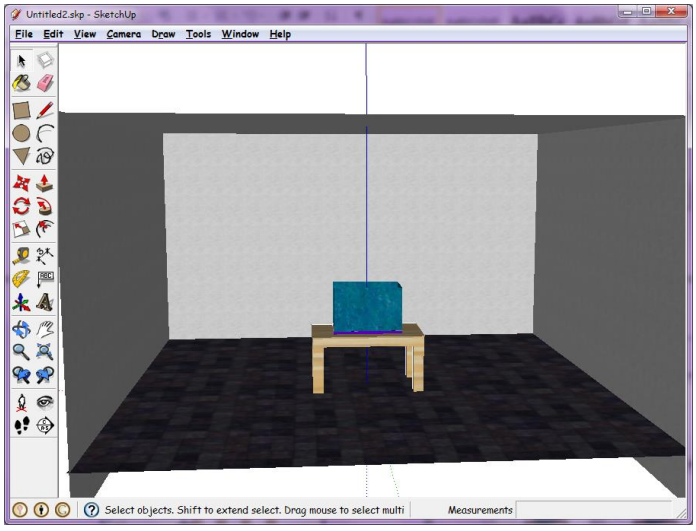
Επιλέγοντας το Paint Bucket εμφανίζεται ένα παράθυρο με όνομα Materials.

Μπορούμε να επιλέξουμε έτοιμες υφές για τα αντικείμενα μας, ή μπορούμε να προσθέσουμε δικές μας. Πατώντας το κουμπί Create Material μας εμφανίζεται το δεύτερο παράθυρο της Εικόνας 35.

Με το κουμπί Browse for material image file, ανοίγουμε ένα παράθυρο από το οποίο μπορούμε να επιλέξουμε την εικόνα που θέλουμε να τοποθετήσουμε στο αντικείμενο.



Εικόνα 35 – materials του SketchUp



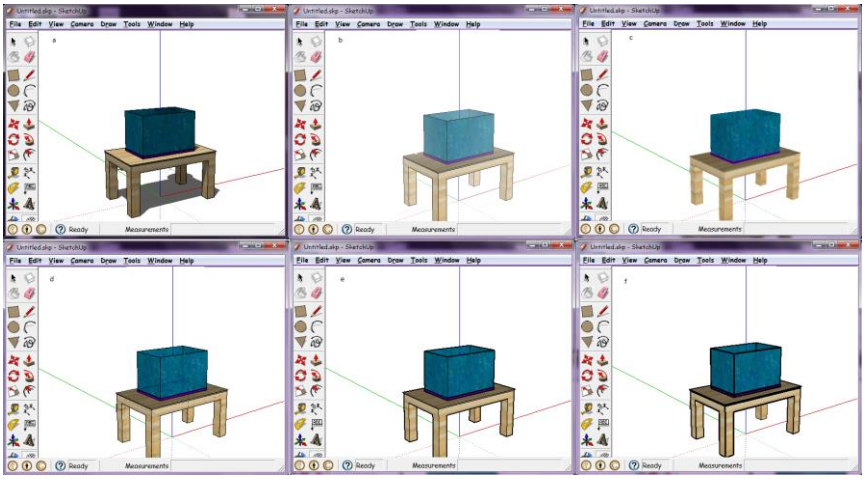
Εικόνα 36 – αποτέλεσμα με την προσθήκη materials στα objects

A.2.2 Styles

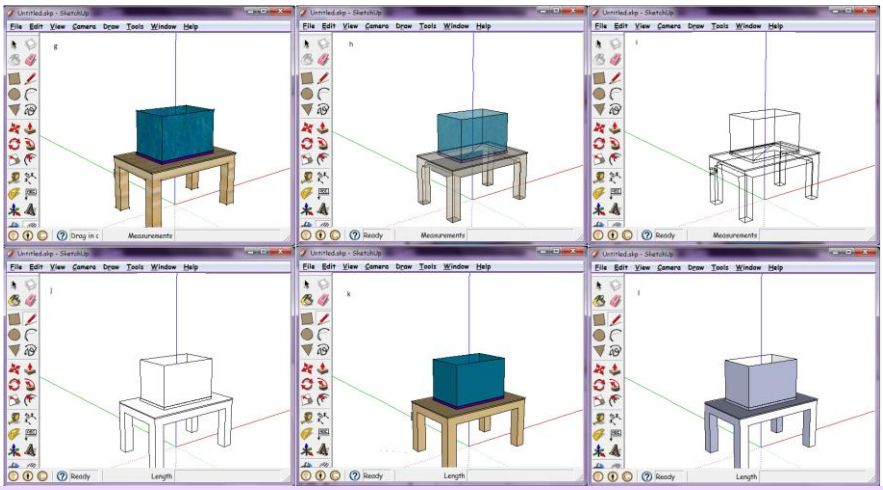
Το μενού View περιέχει επίσης κάποιες επιλογές οι οποίες είναι χρήσιμες για το σχεδιασμό του αντικειμένου μας. Στις εικόνες που ακολουθούν φαίνονται τα αποτελέσματα τους.

a. Η **View→Shadows**: εμφανίζει τη σκιά των αντικειμένων
b. Η **View→Fog** κάνει: τα αντικείμενα θαμπά

c. Η **View→Edge Style→Edges**: αν είναι τσεκαρισμένο φαίνονται οι γραμμές γύρω από τα αντικείμενα.
d. Η **View→ Edge Style→Back Edges**: εμφανίζει και τις πίσω γραμμές διακεκομμένες.
e. Η **View→ Face Style→Wireframe**: εμφανίζει μόνο τις ακμές του αντικείμενου, όπως το X-ray αλλά σε ασπρόμαυρο
f. Η **View→ Face Style→Hidden Line**: εμφανίζει μόνο τις ακμές του αντικειμένου.
g. Η **View→ Face Style→Shaded**: εμφανίζει τα βασικά χρώματα που βάλαμε στο αντικείμενο, χωρίς τις υφές.
h. Η **View→ Face Style→ Monochrome**: εμφανίζει το αντικείμενο ασπρόμαυρο.



Εικόνα 37 – δοκιμές από το μενού View



Εικόνα 38 - δοκιμές από το μενού View

B. Import – Export

B.1 Εξαγωγή από το SketchUp

Για να εξάγουμε ενα αρχείο από το στο Sketch Up πατάμε από το μενού **File→Export**. Μας εμφανίζονται 3 υπό-επιλογές, τις οποίες διαλέγουμε, ανάλογα με το αρχείο εξαγωγής που θέλουμε να δημιουργήσουμε:

- 3D Model

Για εξαγωγή ενός τρισδιάστατου περιβάλλοντος

- 2D Graphic

Για την εξαγωγή μίας δισδιάστατης εικόνας

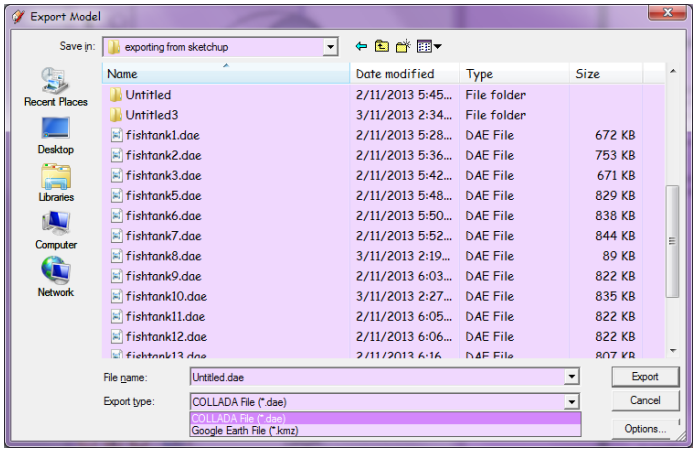
- Animation

Για την εξαγωγή κινουμένων σχεδίων

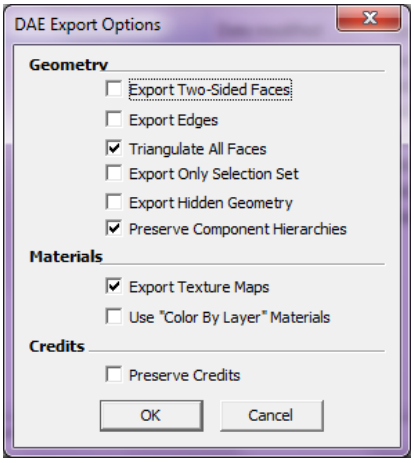
B.1.1 Εξαγωγή τρισδιάστατου μοντέλου

File→Export→3D Model

Όπως φαίνεται στην Εικόνα 39, μπορούμε να εξάγουμε ένα τρισδιάστατο μοντέλο σε δύο τύπους αρχείων. Ο ένας είναι με τη μορφή Collada File (.dae) και ο άλλος με την Google Earth File (.kmz). Το αρχείο .kmz που θα δημιουργήσουμε, μπορούμε να το ανοίξουμε με την εφαρμογή Google Earth η οποία θα το τοποθετήσει σε ένα σημείο που επιθυμούμε.



Εικόνα 39 –τύποι εξαγωγής 3D μοντέλου



Εικόνα 40 – επιλογές εξαγωγής .dae

Το αρχείο .dae, με το οποίο θα ασχοληθούμε, είναι ο μόνος τύπος εξαγωγής που διαθέτει το SketchUp, ο οποίος μπορεί να χρησιμοποιηθεί και από άλλες τρισδιάστατες εφαρμογές όπως το Blender. Στην επιλογή Options εμφανίζεται το παράθυρο της Εικόνας 40.
Στις επιλογές:
Geometry: [58]

- **Export two-sided faces**: διπλασιάζει τον αριθμό των πολυγώνων του αρχείου που εξάγεται, καθώς εξάγει μία όψη για την μπροστινη και μία για την πίσω πλευρά του αντικειμένου, επιβραδύνοντας έτσι την απόδοσή του
- **Export edges**: η επιλογή αυτή χρησιμοποιείται για την διατήρηση των ορατών ακμών ενός μοντέλου
- **Triangulate all faces**: με αυτή την επιλογή όλες οι επιφάνειες του αντικειμένου, διαιρούνται σε τρίγωνα, επειδή κάποιες τρισδιάστατες εφαρμογές υποστηρίζουν μόνο τριγωνισμένες όψεις
- **Export only selection set**: εξάγει επιλεγμένη γεωμετρία στην περιοχή σχεδίασης
- **Export hidden geometry**: εξάγει την κρυμμένη γεωμετρία που υπάρχει μέσα σε ένα μοντέλο
- **Preserve component hierarchies**: διατηρεί την ιεραρχία των στοιχείων

Materials:

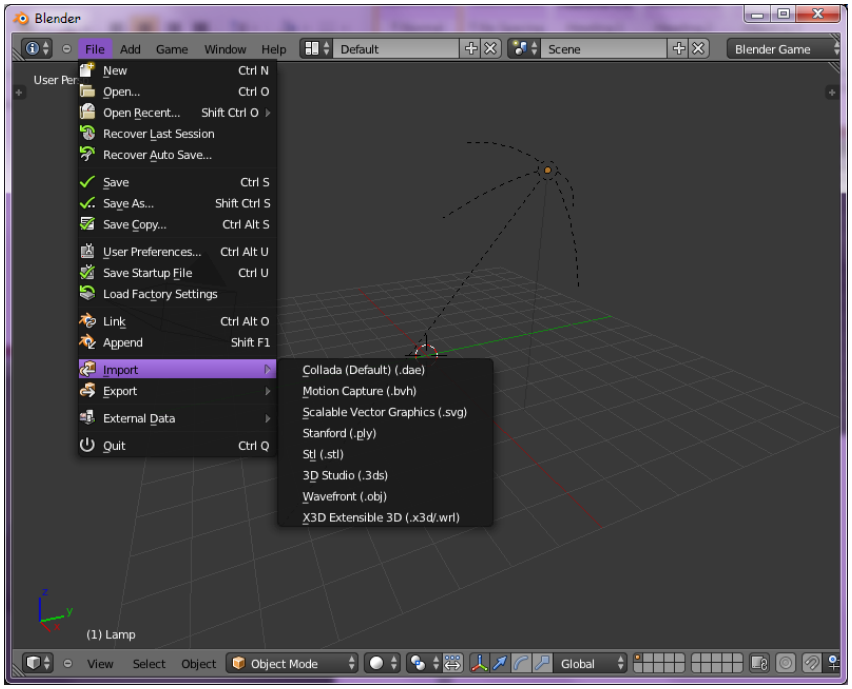
- **Export texture maps**: επιλογή για να εξάγουμε τις υφές στο αρχείο μας
- **Use "color by layer" materials**: εξάγει τα υλικά της γεωμετρίας, βάσει των αναθέσεων του μοντέλου του SketchUp

B.2 Εισαγωγή στο Blender

Έχουμε πλέον δημιουργήσει και εξάγει το αρχείο, από το SketchUp και είναι σειρά να επεξεργαστεί από το Blender.

Για να εισάγουμε ένα αρχείο στο Blender επιλέγουμε από το μενού **File→ Import** όπου μας εμφανίζει τις εξής επιλογές:

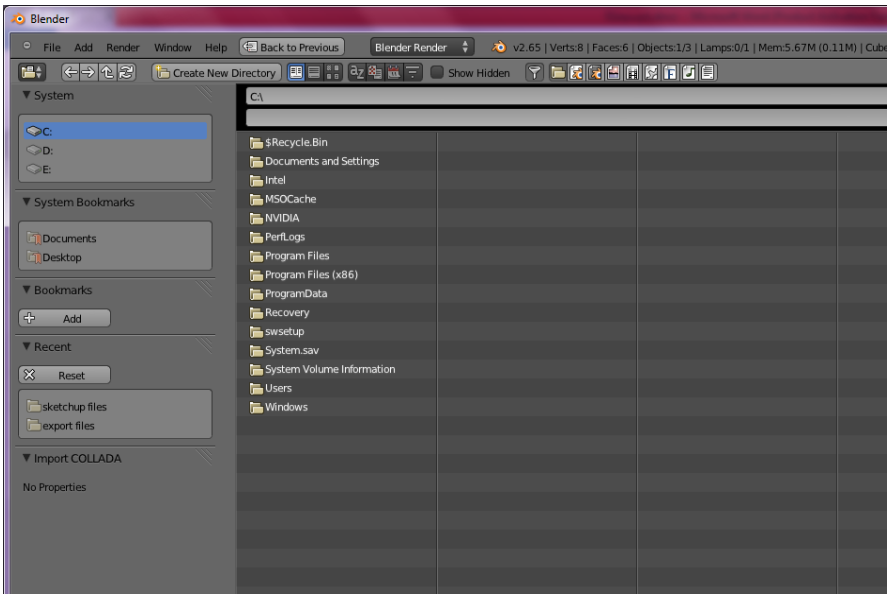
- COLLADA (.dae)
- Motion Capture (.bvh)
- Scalable Vector Graphics (.svg)
- Standford (.ply)
- Stl (.stl)
- 3D Studio (.3ds)
- Wavefront (.obj)
- X3D Extensible (.x3d)/(.wrl)



Εικόνα 41 – τύποι εισαγωγής 3D μοντέλου στο Blender

Επιλέγουμε το Collada αφού αυτόν τον τύπο αρχείου χρησιμοποιήσαμε για την εξαγωγή από το SketchUp.

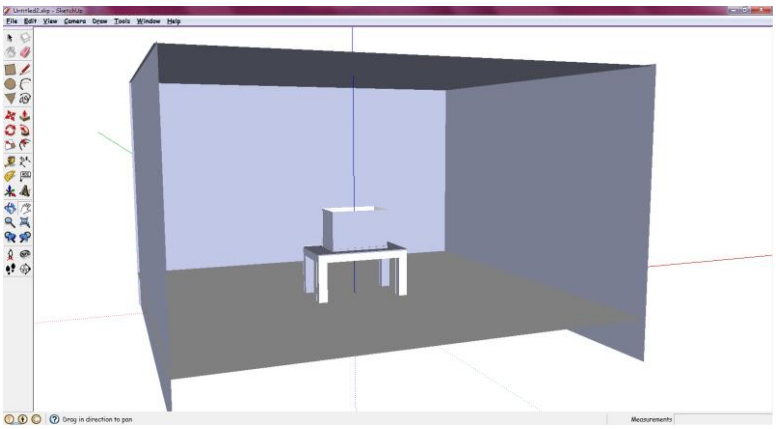
Αφού το επιλέξουμε, ανοίγει ένα παράθυρο στο Blender όπου μας εμφανίζει τους φακέλους αρχείων του υπολογιστή μας. Επιλέγουμε το φάκελο στον οποίο έχουμε αποθηκεύσει το εξαγόμενο αρχείο, το τσεκάρουμε και πατάμε **Import→Collada**. Αμέσως μας το ανοίγει, και πλέον μπορούμε να το επεξεργαστούμε και να το αποθηκεύσουμε ως .blend



Εικόνα 42 – επιλογή του αρχείου .dae που θέλουμε να εισάγουμε

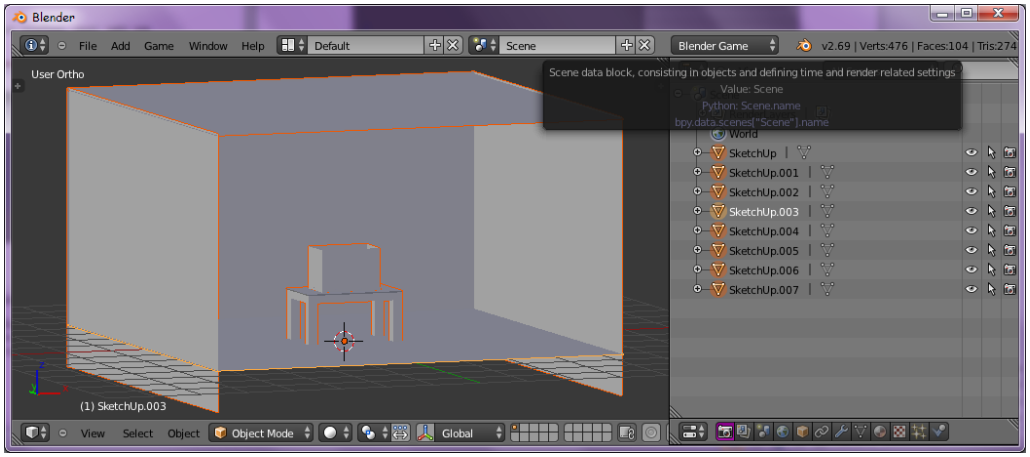
B.3 Αποτελέσματα

Κάνουμε Export από το SketchUp το μοντέλο που φτιάξαμε, χωρίς υφές και χρώματα, με τις προεπιλεγμένες ρυθμίσεις του Options.



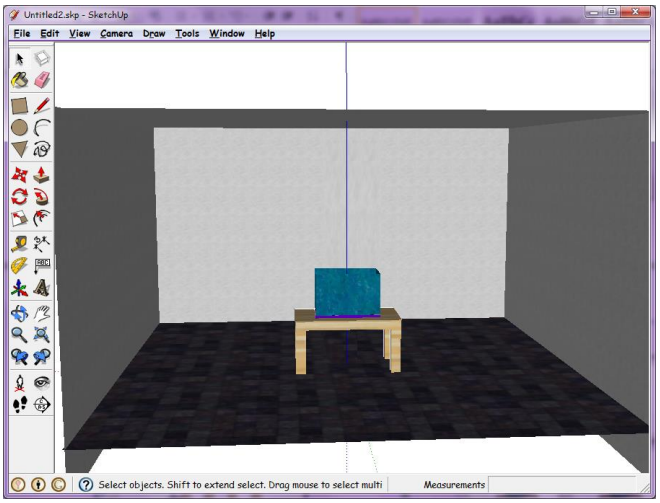
Εικόνα 43 – μοντέλο στο SketchUp για εξαγωγή 1

Αφού κάνουμε Import το μοντέλο στο Blender (Εικόνα 44), παρατηρούμε στο παράθυρο Outliner που μας δείχνει τα αντικείμενα, ότι τα δύο πρώτα με τα ονόματα SketchUp, SketchUp.001 κλπ αναφέρονται με τη σειρά στη γυάλα, στη βάση της, στο τραπέζι, στο πάτωμα και στους τοίχους του δωματίου.

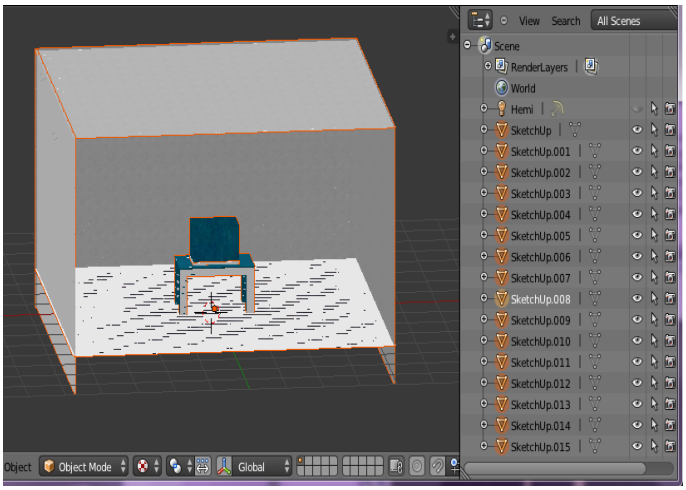


Εικόνα 44 –μοντέλο που κάναμε εισαγωγή στο Blender 1

Επίσης παρατηρούμε ότι, αν βάλουμε υφές στο μοντέλο (Εικόνα 45) και ξανακάνουμε την διαδικασία εισαγωγής/εξαγωγής τα αντικείμενα στο Blender έχουν αυξηθεί (Εικόνα 46).



Εικόνα 45 - μοντέλο στο SketchUp για εξαγωγή 2



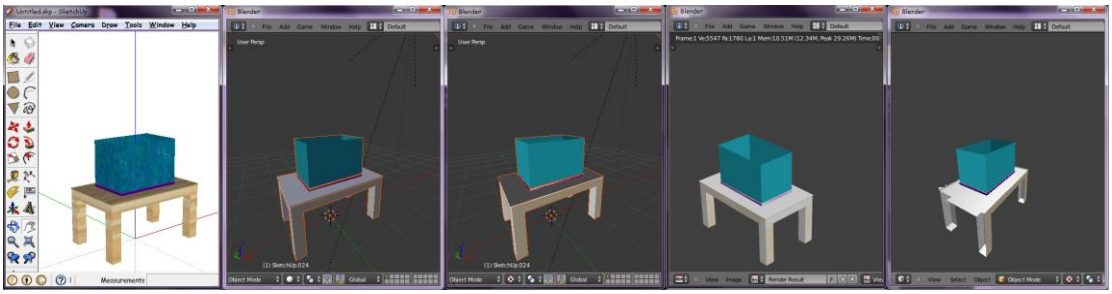
Εικόνα 46 - μοντέλο που κάναμε εισαγωγή στο Blender 2

Κάθε φορά που εξάγουμε από το SketchUp ένα αρχείο στο οποίο έχουμε βάλει υφές, αυτόματα δημιουργείται ένας φάκελος με το όνομα του αρχείου, ο οποίος περιέχει μέσα τις εικόνες των υφών που χρησιμοποιήσαμε σε μορφή .jpg.

Επίσης κάναμε κάποιες δοκιμές με υφές και με διάφορες επιλογές όψης και στιλ του μοντέλου και από την επιλογή Options που βρίσκεται στο Export, και κάθε αρχείο που δημιουργούσαμε το εισάγαμε στο Blender. Παρακάτω αναφέρονται πιο αναλυτικά οι ρυθμίσεις και τα αποτελέσματα.

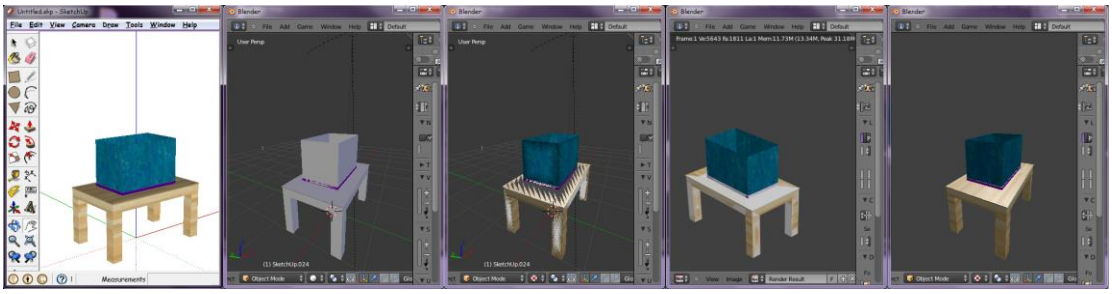
Σημ.: Σε κάθε εικόνα στην οποία φαίνονται οι δοκιμές, η σειρά των στιγμιότυπων είναι η εξής από αριστερά προς τα δεξιά: από το SketchUp με τις ρυθμίσεις που του κάναμε, στο Blender τα αποτελέσματα σε Solid Mode, στο Blender σε Texture Mode, Render Result και Start της Game Engine.

a. Το μοντέλο που εξάγαμε από το SketchUp, είχε τις ρυθμίσεις **Face Style→Shaded with Textures** και **Edge Style→No edges**. Στην επιλογή **Options** δεν έχουμε επιλέξει τίποτα.



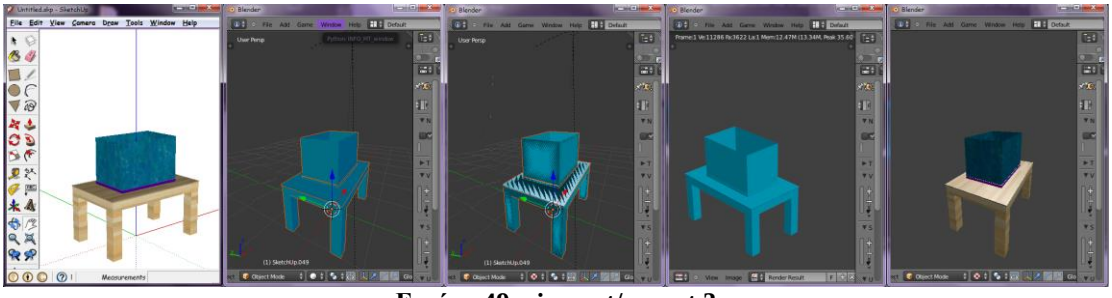
Εικόνα 47 – import/export 1

b. Ρυθμίσεις στο SketchUp, **Face Style→Shaded with Textures** και **Edge Style→No edges**. Στην επιλογή **Options** τσεκάρουμε το **Export Texture Map**.



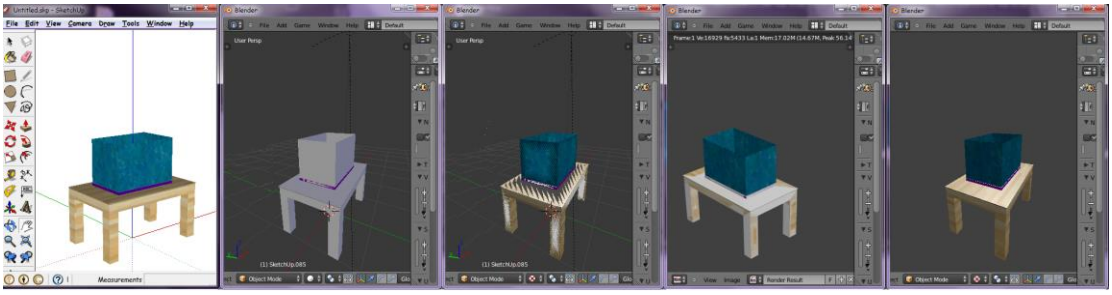
Εικόνα 48 – import/export 2

c. Ρυθμίσεις στο SketchUp, **Face Style→Shaded with Textures** και **Edge Style→No edges**. Στην επιλογή **Options** τσεκάρουμε **Use Color by Material**.



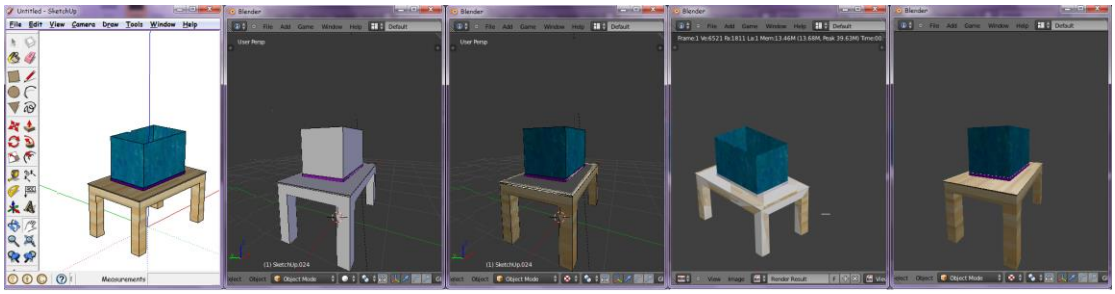
Εικόνα 49 – import/export 3

d. Ρυθμίσεις στο SketchUp, **Face Style→Shaded with Textures** και **Edge Style→No edges**. Στην επιλογή **Options** τσεκάρουμε το **Export Texture Map** και το **Preserve Credits**.



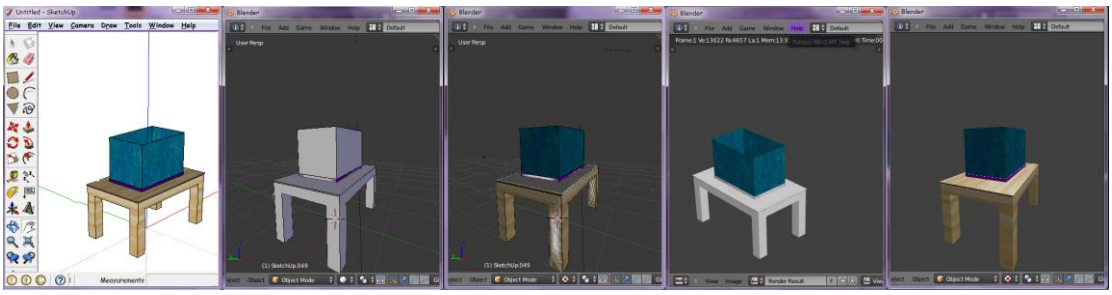
Εικόνα 50 – import/export 4

e. Ρυθμίσεις στο SketchUp, **Face Style→Shaded with Textures** και **Edge Style→Edges**. Στην επιλογή **Options** τσεκάρουμε το **Export two sided Faces**, **Export Edges, Export Texture Map** και το **Preserve Credits**.



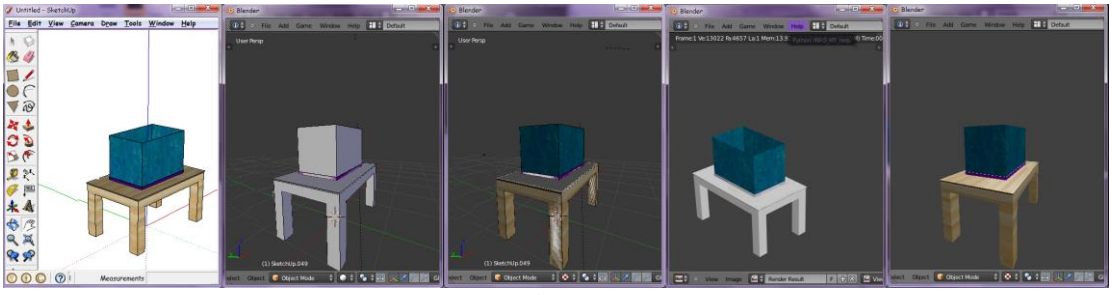
Εικόνα 51 – import/export 5

f. Ρυθμίσεις στο **SketchUp, Face Style→Shaded with Textures** και **Edge Style→Edges.** Στην επιλογή **Options** τσεκάρουμε το **Export Texture Map,** το **Preserve Credits, Triangulate all Faces.**



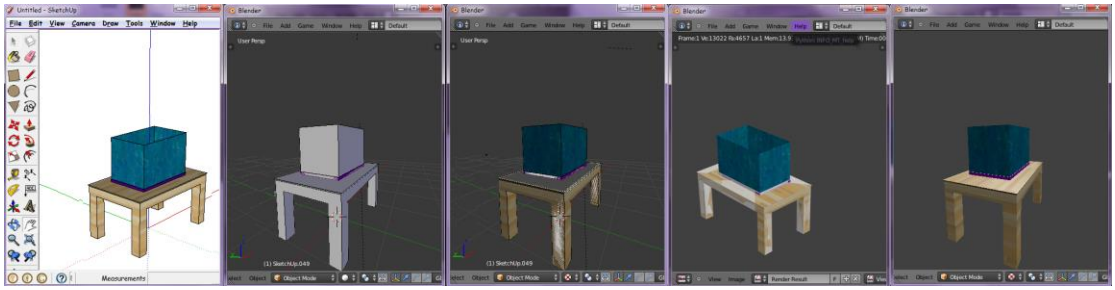
Εικόνα 52 - import/export 6

g. Ρυθμίσεις στο SketchUp**, Face Style→Shaded with Textures** και **Edge Style→Edges**. Στην επιλογή **Options** τσεκάρουμε το **Export Texture Map**, **Export Hidden Geometry** και το **Preserve Credits**.



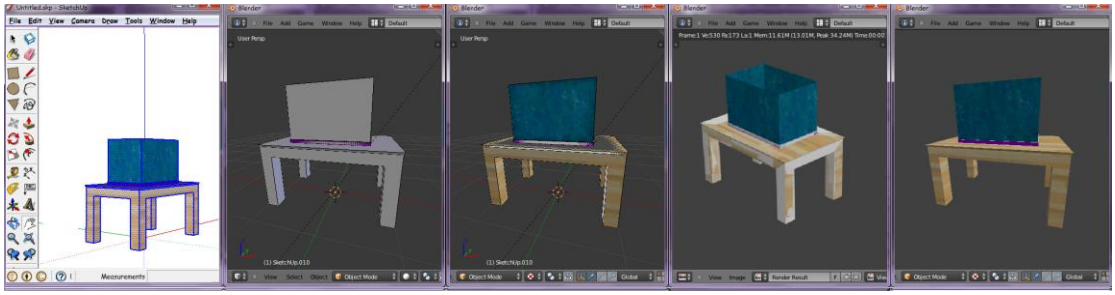
Εικόνα 53 - import/export 7

h. Ρυθμίσεις στο SketchUp, **Face Style→Shaded with Textures** και **Edge Style→Edges**. Στην επιλογή **Options** τσεκάρουμε το **Export two sided Faces**, **Export Edges**, **Export Hidden Geometry**, **Export Texture Map**, και το **Preserve Credits**.



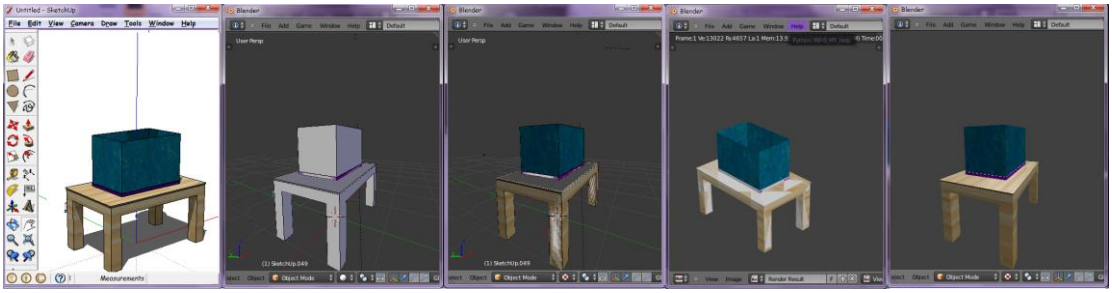
Εικόνα 54 - import/export 8

i. Αρχικά επιλέγουμε με το Select Tool ποιά αντικείμενα του μοντέλου θέλουμε να εξάγουμε. Ρυθμίσεις στο SketchUp, **Face Style→Shaded with Textures** και **Edge Style→No edges**. Στην επιλογή **Options** τσεκάρουμε το **Export two sided Faces**, **Export Edges**, **Export Hidden Geometry**, **Export Texture Map**, το **Preserve Credits** και το **Export Only Selection Set**.



Εικόνα 55 - import/export 9

j. Ρυθμίσεις στο SketchUp **View→Shadows, Face Style→Shaded with Textures** και **Edge Style→Edges.** Στην επιλογή **Options** τσεκάρουμε το **Export two sided Faces, Export Edges, Export Hidden Geometry, Export Texture Map,** και το **Preserve Credits.**

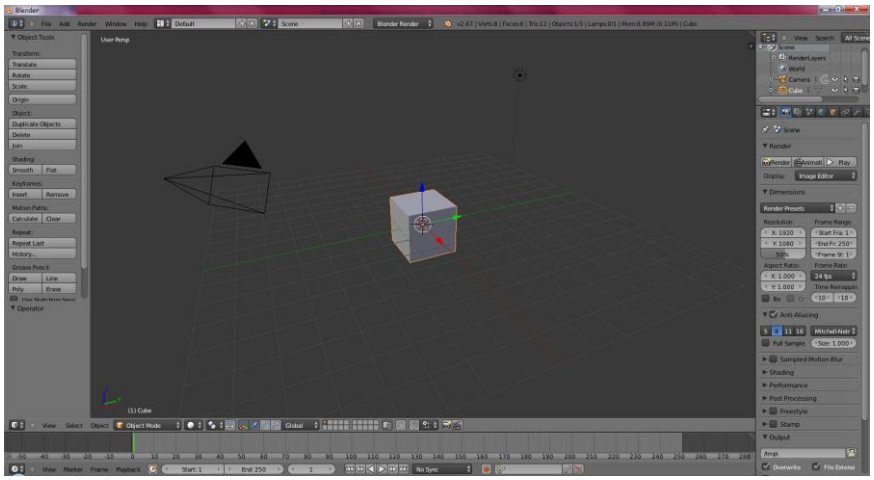


Εικόνα 56 - import/export 10

C. Blender

C.1 Βασικά εργαλεία

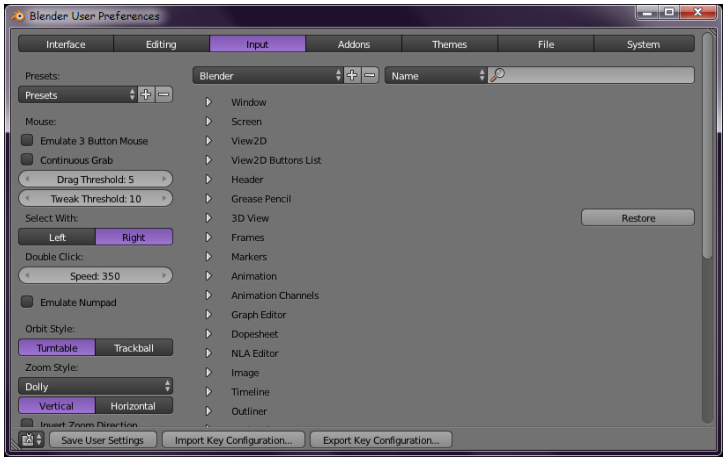
Το interface του Blender εμφανίζεται στο παράθυρο της Εικόνας 57.



Εικόνα 57 – Blender interface

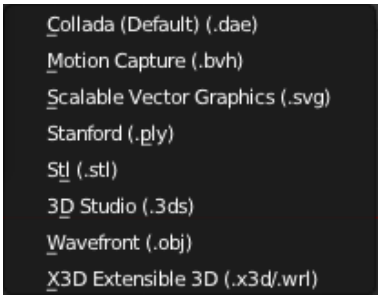
Ψηλά στην οθόνη, έχουμε τον **Info Editor** ο οποίος περιέχει το μενού για την επεξεργασία των αρχείων, για την εισαγωγή αντικειμένων στη σκηνή, ρυθμίσεις για την αναπαράσταση της σκηνής, για το παράθυρο της εφαρμογής και βοήθεια.

Το Blender επιτρέπει στον χρήστη του, να διαμορφώσει όπως θέλει αυτός το interface, τα χρώματα, τα hotkeys, τις κινήσεις του ποντικιού. Απο το μενού **File→ User Preferences**, ανοίγει το παράθυρο της Εικόνας 58, όπου μπορούμε να ρυθμίσουμε τα παραπάνω.

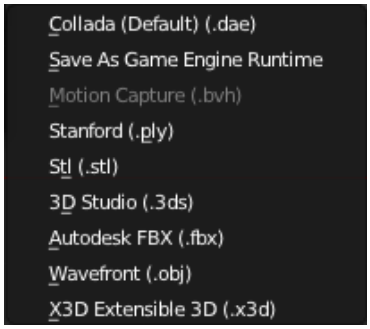


Εικόνα 58 – User Preferences

Μπορούμε να εξάγουμε και να εισάγουμε αρχεία από το Blender. Στην Εικόνα 59 έχουμε τους τύπους εισαγωγής (import) και στην Εικόνα 60 τους τύπους εξαγωγής (export). Οι τύποι αρχείων που θα χρησιμοποιήσουμε, είναι τα Collada (.dae), 3D Studio (.3ds), Wavefront (.obj) για το importing και για το exporting, ενώ επιπλέον για το exporting θα χρησιμοποιήσουμε το Save As Game Engine Runtime.



Εικόνα 59 – import types



Εικόνα 60 – export types

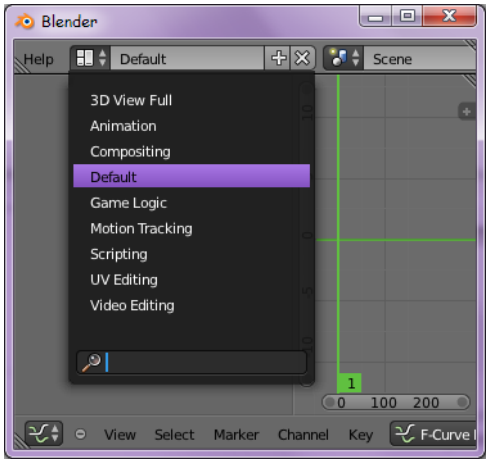
- Το Collada όπως είπαμε και σε παραπάνω ενότητα είναι μια συνεργατική δραστηριότητα σχεδιασμού για τη θέσπιση μιας μορφής ανταλλαγής αρχείων για διαδραστικές τρισδιάστατες εφαρμογές. Τα Collada έγγραφα που περιγράφουν ψηφιακά στοιχεία, είναι αρχεία XML και συνήθως ταυτίζονται με την επέκταση αρχείου .dae [17]
- Το .obj είναι ένας γεωμετρικός ορισμός για μορφή αρχείου. Αναπτύχθηκε για πρώτη φορά από τη Wavefront Technologies για το Advanced Visualizer ένα πακέτο λογισμικού για 3d γραφικά και animation. Αυτός ο τύπος αρχείου είναι ανοιχτός και έχει υιοθετηθεί από άλλους προμηθευτές τρισδιάστατων εφαρμογών γραφικών. Είναι μια παγκοσμίως αποδεκτή μορφή δεδομένων που αναπαριστά τρισδιάστατη γεωμετρία, τη θέση κάθε κορυφής, τη UV θέση κάθε υφής συντεταγμένων κορυφών, και τις επιφάνειες που κάνουν κάθε πολύγωνο να ορίζεται ως μια λίστα κορυφών. Οι συντεταγμένες του .obj δεν έχουν μονάδες, έτσι τα εν λόγω αρχεία μπορούν να περιέχουν πληροφορίες κλίμακας που να είναι αναγνωρίσιμες από τον άνθρωπο. [59]
- Το .3ds είναι μια από τις μορφές αρχείων που χρησιμοποιούνται από το Autodesk 3D Studio Max, ένα λογισμικό για 3D σχεδιασμό, animation και απόδοση.
- Το Save As Game Engine Runtime λειτουργεί αποκλειστικά στη Game Engine του Blender και εξάγει το παιχνίδι που έχει δημιουργηθεί με τις κινήσεις, τις υφές και όλα όσα του έχουμε βάλει.

Το μενού Add, διαθέτει τα αντικείμενα, κάμερες, λάμπες, οπλισμούς και ότι άλλο χρειάζεται για να συνθέσουμε μία σκηνή. Απο το **Add→Mesh** μπορούμε να βρούμε αντικείμενα τύπου σφαίρας, κύβου, κώνου, κυλίνδρου, κ.α.. Από το **Add→Camera** προσθέτουμε μία κάμερα. **Add→Lamp** λάμπες ανάλογα με το είδος που θέλουμε. **Add→Armature** βάζουμε οπλισμό σε κάποιο αντικείμενο. Επίσης υπάρχει και το **Add→Empty→Plain Axis** το οποίο προσθέτει απλά τους 3 άξονες σε ένα αντικείμενο.

Το μενού Help μπορούμε να βρούμε διάφορα χρήσιμα tutorial για την εκμάθηση του Blender. Επίσης από το **Help→Splash Screen** ανοίγει το παράθυρο το οποίο εμφανίζεται και όταν ανοίγουμε την εφαρμογή. Στα αριστερά του παραθύρου έχουμε ξανά τους συνδέσμους για το site του Blender και tutorial. Στα δεξιά έχουμε τα πιο πρόσφατα αρχεία που έχουμε ανοίξει.

Εικόνα 61 – splash screen

Στην Εικόνα 62 μπορούμε να επιλέξουμε με ποιά μορφή θέλουμε να επεξεργαστούμε τη σκηνή μας, δηλαδή αν θέλουμε να φτιάξουμε Animation θα διαλέξουμε την αντίστοιχη επιλογή. Έπειτα, βλέπουμε μία λίστα με τις σκηνές τις οποίες έχουμε και μπορούμε από εκεί να τις επεξεργαστούμε.



Εικόνα 62 - Blender Views

Τέλος στη λίστα στο πλάι, έχουμε τις τρείς μηχανές στις οποίες μπορούμε να δουλέψουμε στο Blender, δηλαδή τις Blender Render, Blender Game και Cycles Render.

- **Blender Render**: είναι η τελική αναπαράσταση όπου δημιουργείται η εικόνα η οποία αναπαριστά την τρισδιάστατη σκηνή μας. Περιέχει γραμμή σάρωσης, αντιγραφή της ακτίνας φωτός, έμμεσο φωτισμό και απόφραξη περιβάλλοντος που μπορεί να εξάγει σε μεγάλη ποικιλία μορφών [60]
- **Blender Cycles**: αντιγράφει την ακτίνα φωτός και κάνει διαδραστικό rendering, έχει δικό του σύστημα σκίασης και δημιουργίας υφών και επιτάχυνση GPU [61]
- **Blender Game**: προσφέρει τη δημιουργία διαδραστικων τρισδιάστατων εφαρμογών ή προσομοιώσεων. Κάνει συνεχή απόδοση σκηνών σε πραγματικό χρόνο και προσφέρει επίσης στον χρήστη δυνατότητες αλληλεπίδρασης κατά τη διάρκεια της διαδικασίας απόδοσης. [62] [63]

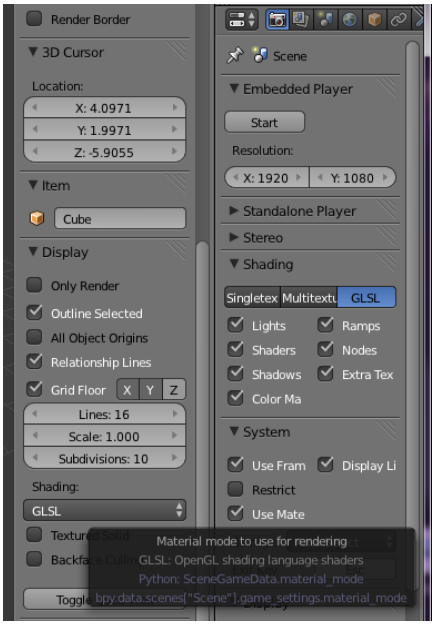
Αντί να διαλέγουμε κάθε φορά ποιούς Editors θέλουμε να χρησιμοποιήσουμε, το Blender έχει έτοιμα κάποια interfaces. Αυτά βρίσκονται στην πρώτη λίστα του Info Editor. Αυτά που θα χρησιμοποιήσουμε εμείς είναι τα Animation, Default, Game Logic, Scripting, UV Editing. Κάθε ένα από αυτά έχει προεπιλεγμένους κάποιους Editors, τους οποίους μπορούμε πάντα να αλλάξουμε ανάλογα με τις ανάγκες μας.

C.2 Blender Game Engine

Εμείς θα ασχοληθούμε με τη μηχανή Game Engine και θα συνεχίσουμε την περιγραφή του Blender σε αυτήν αφού όλη η επεξεργασία του μοντέλου θα γίνει εκεί.

Αλλάζουμε λοιπόν απο τη λίστα που αναφέραμε παραπάνω, το Blender Render σε Blender Game.

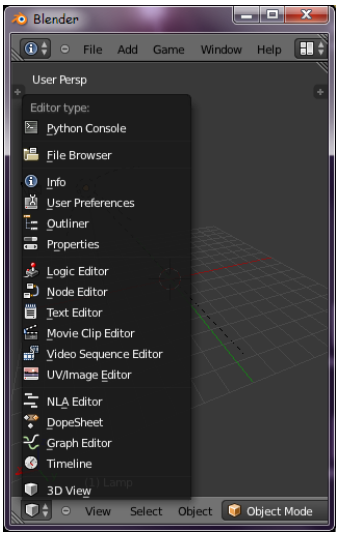
Και οι δύο αυτοί τρόποι φαίνονται στην Εικόνα 63.



Εικόνα 63 - GLSL Shading

Στο interface της εφαρμογής δεν αλλάζει τίποτα από πριν. Παρατηρούμε λοιπόν ότι χωρίζεται σε 5 μέρη. Σε 5 διαφορετικούς editors τους Info, 3D View, Outliner, Properties, Timeline.

Τους Editors αυτούς, μπορούμε να τους αλλάξουμε, ανάλογα με το τι μας εξυπηρετεί να κάνουμε κάθε φορά. Όπως φαίνεται στην Εικόνα 64 πατώντας πάνω σε ένα εικονίδιο ενός Editor μας εμφανίζει και τους υπόλοιπους που διαθέτει το Blender. Αυτοί που θα χρησιμοποιήσουμε εμείς κυρίως είναι οι Info, 3D View, Outliner, Properties, UV/Image, Logic, Text, Timeline, Graph, DopeSheet, Python Console.



Εικόνα 64 – Blender Editors

- **Info Editor** εκτός των παραπάνω που αναφέρθηκαν, διαθέτει ένα πλαίσιο στο οποίο συμπληρώνεται αυτόματα σε γλώσσα Python οποιαδήποτε ενέργεια κάνουμε στο Blender
- **3D View Editor** ο οποίος καταλαμβάνει το μεγαλύτερο μέρος της οθόνης, περιέχει την σκηνή στην οποία δημιουργούμε τα μοντέλα μας. Υπάρχουν προεπιλεγμένα μια κάμερα, μία λάμπα και ένας κύβος. Ακριβώς από κάτω από την σκηνή υπάρχει ένα μενού. Στο View δείχνει κάποια εργαλεία για την επεξεργασία των αντικειμένων, όπως αυτό στα αριστερά της οθόνης, όψεις από τις οποίες μπορούμε να δούμε το αντικείμενο. Το Select μπορούμε να επιλέξουμε τα αντικείμενα, τις κάμερες να κάνουμε αντιστροφή των επιλεγμένων. Επίσης περιέχει κάποια χρήσιμα εργαλεία επιλογής. Στο Object έχουμε επιλογές για το αντικείμενο, όπως διαγραφή, μετακίνηση, ρυθμίσεις

κίνησης, μεταφορά σε άλλο layer. Δίπλα στο πλαίσιο έχει τις καταστάσεις (Modes) με τις οποίες μπορούμε να επεξεργαστούμε το αντικείμενο. Ακριβώς δίπλα έχει τις καταστάσεις σκίασης τις οποίες μπορούμε να εναλλάσσουμε, ανάλογα τη δουλεία που θέλουμε να κάνουμε (Viewport Shading), το σημείο στο οποίο θα είναι ο άξονας περιστροφής του αντικειμένου (Pivot Point), τα εργαλεία για τον τρισδιάστατο χειρισμό του αντικειμένου με βάση τους άξονες, καθώς και τον προσανατολισμό των αξόνων. Τα κουτάκια δίπλα, είναι τα layer τα οποία περιέχει μία σκηνή και βοηθάνε στον έλεγχο και την οργάνωση μίας σκηνής

- **Outliner** δείχνει ένα δέντρο με την ιεραρχία των αντικειμένων που βρίσκονται σε κάθε σκηνή. Οτι και να προσθέσουμε σε ένα αντικείμενο, κάποια υφή ή animation, θα φανεί κάτω από αυτό. Επίσης στα δεξιά των αντικειμένων έχουμε τρία εικονίδια. Με το μάτι ρυθμίζουμε αν το αντικείμενο θα φαίνεται στην σκηνή, με τον δείκτη αν το αντικείμενο μπορούμε να το επιλέγουμε και με την κάμερα, αν θα φαίνεται κατά το Rendering ή το Start
- **Properties** δίνει τις ιδιότητες για το Rendering, για την κάθε σκηνή, για τον κόσμο δηλαδή για το γενικό περιβάλλον της σκηνής μας, και για τα αντικείμενα. Απο τις ιδιότητες, μπορούμε να τους αλλάξουμε θέση, μέγεθος, να τα τροποποιήσουμε, να τους βάλουμε χρώματα και υφές και να προσδιορίσουμε την φυσική τους συμπεριφορά
- **UV/Image Editor** χρησιμοποιείται για την απόδοση του μοντέλου σαν εικόνα. Επίσης χρησιμοποιείται για τη δημιουργία υφών, με τη μέθοδο του UV Mapping
- **Logic Editor** είναι το βασικότερο εργαλείο για τη δημιουργία ενός διαδραστικού τρισδιάστατου μοντέλου, ή ενός παιχνιδιού. Σε αυτόν θα προσδιορίσουμε τι ενέργειες θα γίνονται μέσα στη σκηνή μας
- **Text Editor** είναι ένα ενσωματωμένο σημειωματάριο του Blender στο οποίο μπορούμε να γράψουμε python scripts. Για να λειτουργήσει, χρειάζεται ο **Logic Editor**
- **Timeline**, χρησιμεύει για τη δημιουργία keyframe και παρακολούθηση ενός Animation μέσω αυτών
- **DopeSheet Editor** δείχνει πιο λεπτομερώς τα keyframes και μας επιτρέπει να τους κάνουμε αλλαγές, όπως αντιγραφή, μετακίνηση, διαγραφή
- **Graph Editor** δείχνει ένα διάγραμμα ταχύτητας-απόστασης το οποίο καθορίζει την ταχύτητα και την επιτάχυνση του αντικείμενου. Τα σημεία που φαίνονται σε αυτό είναι τα keyframes και μπορούμε να αλλάξουμε την ταχύτητα ενός αντικειμένου, πειράζοντας αυτά
- **Python Console Editor** είναι μία κονσόλα στην οποία γράφουμε εντολές σε γλώσσα Python οι οποίες μπορεί να αναφέρονται στην επεξεργασία του μοντέλου για παράδειγμα και να εκτελούνται αμέσως

C.3 Δημιουργία αντικειμένων

Απο το SketchUp δημιουργήσαμε τα βασικά αντικείμενα του παιχνιδιού μας και μαζί με τα textures κάποιων απο αυτών, τα εισάγαμε στο Blender.

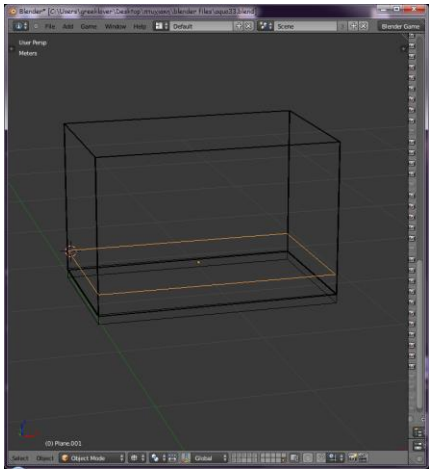
Στο Blender προσθέσαμε κάποια εξτρά αντικείμενα τα οποία κατεβάσαμε από την σελίδα της Turbosquid, πολλά από τα οποία είχαν έτοιμα texture maps. Προσθέσαμε δύο επιπλέον για τη δημιουργία της άμμου και του νερού. Τοποθετήσαμε τις κάμερες με τις οποίες θα προηγούμαστε στο παιχνίδι και τις λάμπες για τον φωτισμό της σκηνής. Επίσης δημιουργήσαμε κάποια αντικείμενα με τη χρήση των καμπυλών Bezier.

a. Meshes

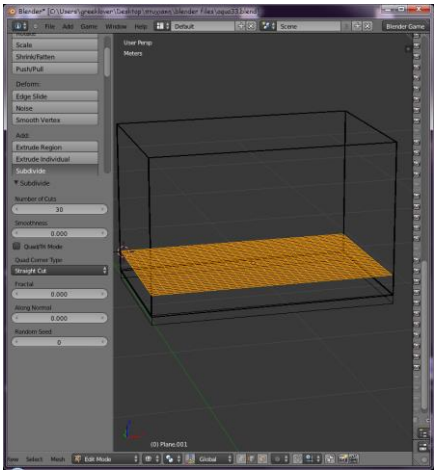
Για τη δημιουργία του εσωτερικού πατώματος του ενυδρείου, θα χρησιμοποιήσουμε ένα Plane, δηλαδή ένα επίπεδο.

Αρχικά θα επιλέξουμε το αντικείμενο το οποίο είναι το ενυδρείο μας, και πατώντας **Shift S→Cursor to Selected** θα βάλουμε τον κέρσορα στην ίδια θέση που είναι και το ενυδρείο. Στη συνέχεια θα πάμε από το μενού **Add→Mesh→Plane** και θα προστεθεί στο σημείο που βρίσκεται ο κέρσορας ένα Plane. Αλλάζουμε το Mode σε Wireframe κάτω στον 3D View Editor, και κάνουμε το μέγεθος του Plane ίσο με το μέγεθος του πάτου του ενυδρείου. Στη συνέχεια έχοντας το Plane επιλεγμένο μεταφερόμαστε σε Edit Mode για να το επεξεργαστούμε ώστε να το κάνουμε να μοιάζει με ένα στρώμα άμμου. Σε Edit Mode πατάμε από την καρτέλα Properties την επιλογή Subdivide και

από κάτω στο Number of Cuts βάζουμε 30. Έτσι το Plane θα χωριστεί σε 30 μικρότερα κομμάτια, τα οποία μπορούμε να επεξεργαστούμε ξεχωριστά το κάθε ένα.

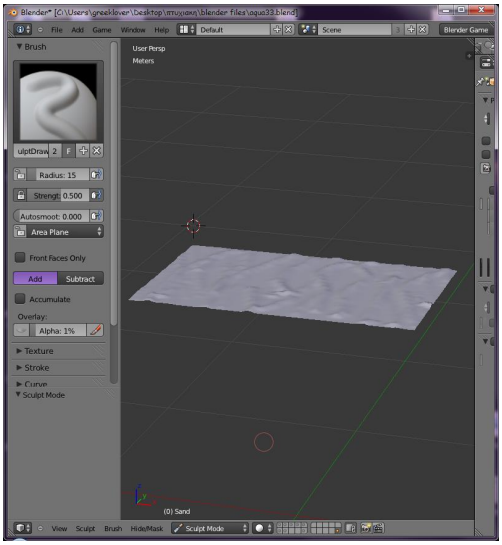


Εικόνα 65 – δημιουργία plane



Εικόνα 66 – επεξεργασία plane σε Edit Mode

Μετά αλλάζουμε από Edit Mode σε Sculpt Mode και από Wireframe σε Solid. Στο Sculpt Mode μπορούμε να συμπεριφερθούμε στο κάθε αντικείμενο σαν ένα γλυπτό. Με το ποντίκι λοιπόν δημιουργούμε εξογκώματα πάνω στο Plane όπως στην Εικόνα 67. Τέλος επιστρέφοντας σε Object Mode πατάμε το κουμπί Smooth από την καρτέλα αριστερά της σκηνής Tool Shelf, ώστε να ομαλοποιήσουμε την "άμμο" μας.



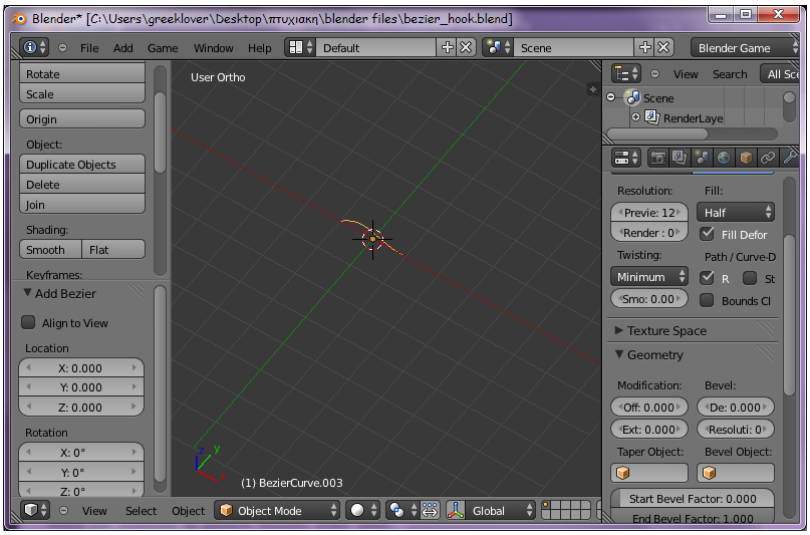
Εικόνα 67 - επεξεργασία plane σε Sculpt Mode

Με τον ίδιο τρόπο δημιουργούμε ένα Plane για το νερό, το φέρνουμε στο ίδιο μέγεθος με αυτό της άμμου, αλλά δεν το επεξεργαζόμαστε σε Edit Mode ή σε Sculpt Mode. Η διαμόρφωση αυτού του Plane, θα αναλυθεί παρακάτω.

b. Bezier Curves

Με τη βοήθεια των καμπυλών Bézier φτιάξαμε τον "παίχτη" του παιχνιδιού μας, ένα αγκίστρι και κάποια αντικείμενα τα οποία έχουν τη μορφή των φυκιών.

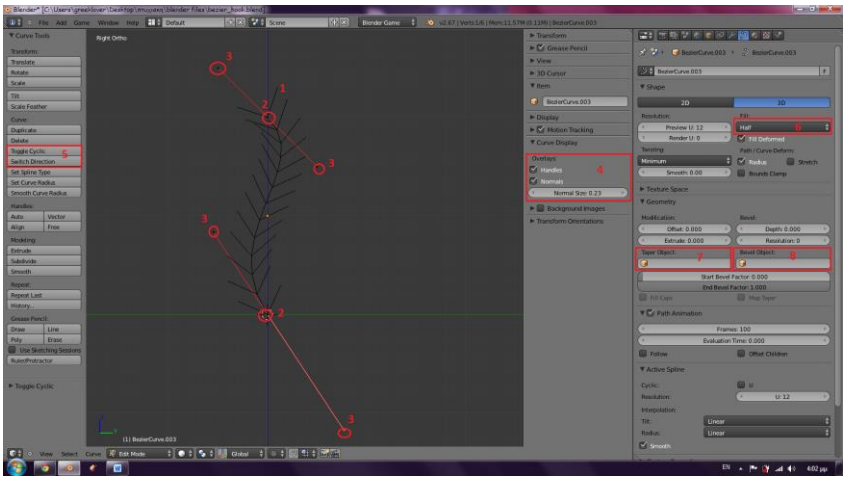
Για τη δημιουργία του αγκιστριού, ανοίξαμε από το μενού Add του Info Editor, την επιλογή **Curve→Bezier**.



Εικόνα 68 – προσθήκη ενός Bézier Curve

Με τις επιλογές rotate, move, scale φέρνουμε την καμπύλη στη θέση που θέλουμε.

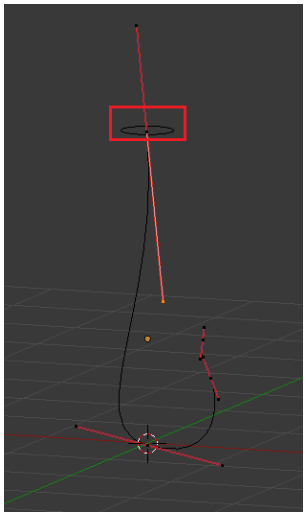
Μεταφερόμενοι σε Edit Mode (Εικόνα 69) μπορούμε να επεξεργαστούμε το σχήμα που θέλουμε να έχει η καμπύλη μας.



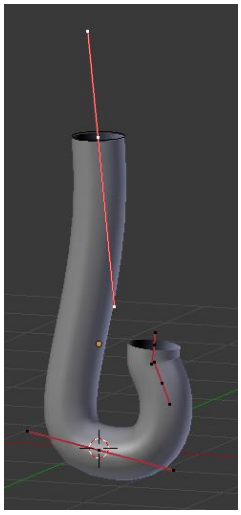
Εικόνα 69 – επεξεργασία του Curve σε Edit Mode

Στο σημείο 1 της εικόνας, βλέπουμε κάποιες γραμμές σαν βέλη πάνω στην καμπύλη, τα οποία καθορίζουν την αρχή και το τέλος της. Τα σημεία 2, είναι η αρχή (κάτω) και το τέλος (πάνω) της καμπύλης. Κάνοντας κλικ πάνω τους και πατώντας **G** μπορούμε να μεγαλώσουμε την καμπύλη μας. Επίσης πατώντας Ctrl και κλικ επεκτείνουμε την καμπύλη από μία κορυφή, σε όποιο σημείο θέλουμε εμείς. Απο τα σημεία 3, με δεξί κλίκ μπορούμε να ορίσουμε την περιστροφή της καμπύλης. Στο 4 η επιλογή Handles απενεργοποιεί τις κόκκινες γραμμές με τις οποίες περιστρέφουμε την καμπύλη, ενώ η Normals απενεργοποιεί τα βέλη που δείχνουν τη φορά της. Στο 5 το κουμπί Toggle Cyclic ενώνει σε κύκλο τις άκρες της καμπύλης, και το Switch Direction αλλάζει τη φορά των βελών. Στο σημείο 6, υπάρχει μία λίστα με επιλογές γεμίσματος της καμπύλης. Το 7 και το 8 είναι τα αντικείμενα τα οποία επηρεάζουν την καμπύλη. Το Bevel Object είναι εκείνο σύμφωνα με το οποίο η καμπύλη θα αποκτήσει ένα σχήμα, το Taper Object είναι εκείνο το οποίο θα διαμορφώσει το σχήμα της.

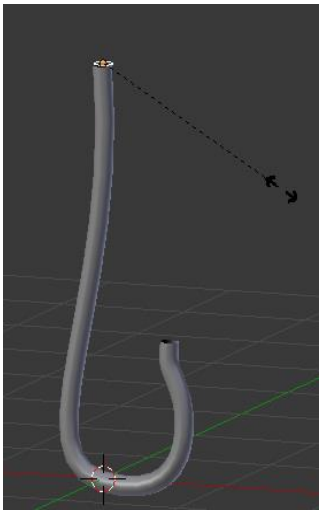
Ακολουθώντας τα παραπάνω φτιάξαμε το βασικό σχήμα για το αγκίστρι. Ο κύκλος που φαίνεται πάνω, είναι ο οδηγός για να γεμίσουμε την καμπύλη. Πατώντας το Bevel Object και επιλέγοντας το όνομα του κύκλου. Σε Object Mode αν μικρύνουμε τον κύκλο, αυτόματα λεπταίνει και ο "σωλήνας" γύρω από την καμπύλη (Εικόνα 70-72).



Εικόνα 70 – βασικό σχήμα Curve



Εικόνα 71 – γέμισμα Curve



Εικόνα 72 – διαμόρφωση Curve

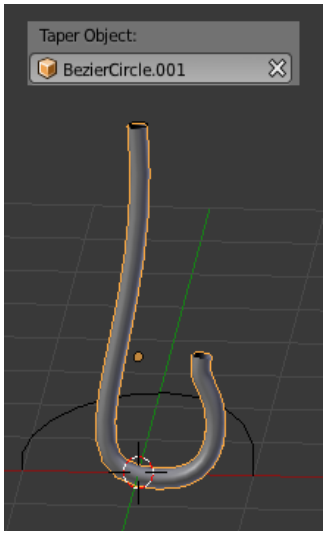
Επίσης στην Εικόνα 71 βλέπουμε ότι μπορούμε ακόμα να περιστρέψουμε την καμπύλη.

Προσθέτουμε μία δεύτερη καμπύλη σε σημείο όπως στην Εικόνα 73. Ενώ έχουμε επιλεγμένη τη βασική μας καμπύλη, διαλέγουμε στο πεδίο Taper Object τη δεύτερη καμπύλη που προσθέσαμε. Επιλέγοντας της και σε Edit Mode μπορούμε πειράζοντας την καμπυλότητά της, να διαμορφώσουμε το σχήμα του αγκιστριού.

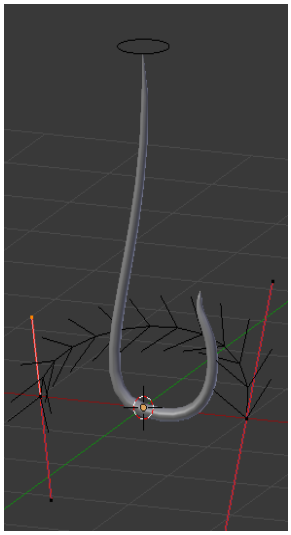
Ακολουθούμε την ίδια διαδικασία, για μία ακόμα καμπύλη, η οποία θα είναι το σκοινί στο οποίο είναι δεμένο το αγκίστρι και ένας κύκλος Bezier τον οποίο μπορούμε να επεξεργαστούμε ακριβώς όπως και τις καμπύλες. Το τελικό αποτέλεσμα είναι στην Εικόνα 75.

Αυτά όμως που έχουμε φτιάξει είναι αντικείμενα τύπου Bezier Curve, για να μπορέσουμε να τα χειριστούμε σαν τα υπόλοιπα, πρέπει να τα μετατρέψουμε σε Mesh Objects. Επιλέγουμε το Curve το οποίο θέλουμε να μετατρέψουμε και πάμε από το μενού **Object→Convert To→Mesh From Curve/Metal/Surf,Text** ή αλλιώς χρησιμοποιώντας το **Alt C.**

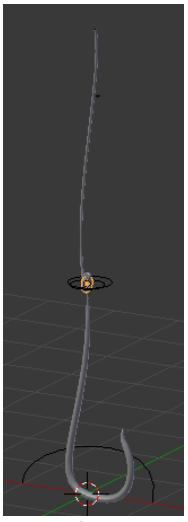
Τέλος κάνουμε parenting όλα τα mesh objects και επιλέγοντας τα, τα κάνουμε Export σαν .Obj, για να μπορέσουμε να τα εισάγουμε στο παιχνίδι.



Εικόνα 73 - διαμόρφωση 1

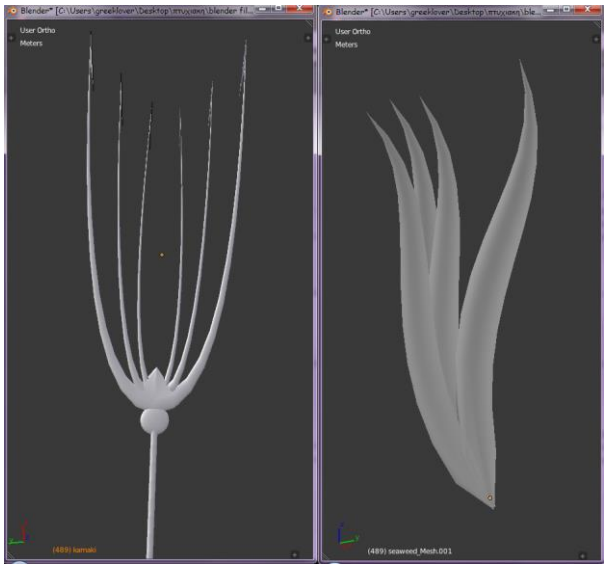


Εικόνα 74 – διαμόρφωση 2



Εικόνα 75 - τελικό αποτέλεσμα

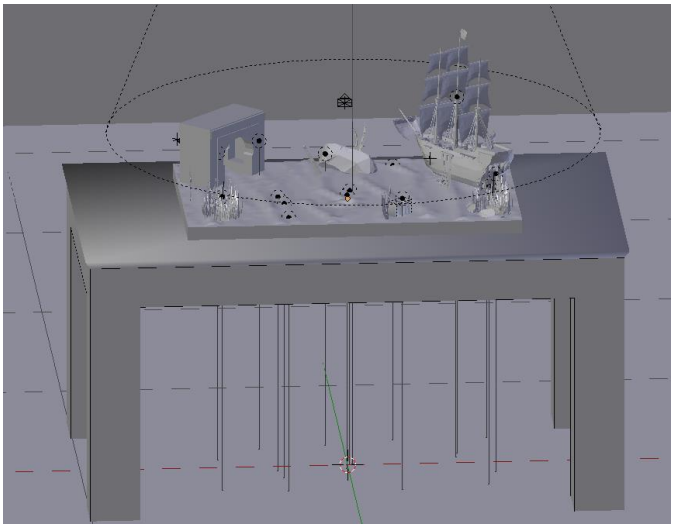
Ακριβώς τη ίδια διαδικασία δημιουργήσαμε το καμάκι και τα φύκια, όπως φαίνονται στην εικόνα 76.



Εικόνα 76 –αντικείμενα σχεδιασμένα με Bezier Curves

Αποτέλεσμα

Τοποθετώντας όλα τα αντικείμενα μαζί, έχουμε το αποτέλεσμα της εικόνας 77. Την γυάλα, τους τοίχους και το ταβάνι τα έχουμε κρύψει για φαίνεται το εσωτερικό τους. Επίσης έχουμε προσθέσει και τις κάμερες σε διάφορα σημεία, ώστε να μπορούμε να προηγηθούμε κατά τη διάρκεια του παιχνιδιού.



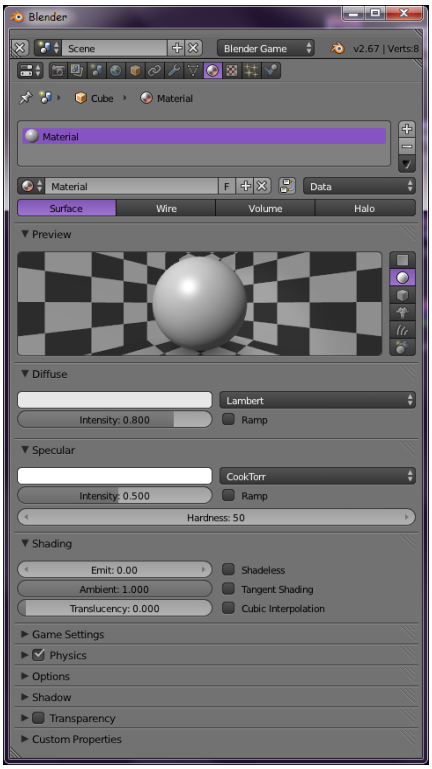
Εικόνα 77 – αποτέλεσμα όλων των objects του παιχνιδιού

Αφού τοποθετήσαμε όλα τα αντικείμενα που συνθέτουν το παιχνίδι μας στη θέση τους, σειρά έχει να τα κάνουμε όσο πιο ρεαλιστικά μπορούμε, προσθέτοντας τους Materials, Textures και βάζοντας σε όσα πρέπει οπλισμό (Armature).

C.4 Materials and Textures

Όπως έχει αναφερθεί σε παραπάνω ενότητα, κάποια απο τα αρχικά αντικείμενα του μοντέλου που εισάγαμε στο Blender, είχαν ήδη υφές από το SketchUp. Τα υπόλοιπα, θα τα μορφοποιήσουμε με τις ρυθμίσεις του Blender, μία διαδικασία η οποία είναι εντελώς διαφορετική από αυτή του SketchUp.

C.4.1 Materials



Εικόνα 78 - Material

Αρχικά με την φράση *Cube>Material* καταλαβαίνουμε ότι το αντικείμενο το οποίο θα επεξεργαστούμε έχει το όνομα Cube.

Στο πλαίσιο απο κάτω υπάρχει μια λίστα με τα material τα οποία έχουμε βάλει στο αντικείμενο μας. Προσθέτουμε και αφαιρούμε απο τα +/ -.

Ορίζουμε ένα όνομα για το κάθε material. Απο κάτω επιλέγουμε πώς θέλουμε να φαίνεται κατα το rendering ή την εκκινηση του παιχνιδιού το αντικείμενο. Surface ειναι η συμπαγής μορφή του αντικειμένου, wire ειναι με μορφή πλέγματος, volume το αντικείμενο εμφανίζεται με βάση τον όγκο του και το halo δείχνει τα σημεία στα οποία βρίσκονται οι κορυφές του αντικείμενου ανάλογα με τη γεωμετρία του, σαν ένα φωτοστέφανο.

Η μορφή που χρησιμοποιείται κυρίως ειναι η Surface.

Στην καρτέλα Preview υπάρχει μια αναπαράσταση του αντικείμενου μετα απο κάθε ρύθμιση που του κάνουμε. Στα κουτάκια δεξιά επιλέγουμε το σχήμα που θέλουμε να έχει.

Στην Diffuse ορίζουμε το χρώμα του, την έντασή (intensity) του και τον τρόπο με τον οποίο θα πέφτει ή σκιά σε αυτό.

Στην καρτέλα Specular ορίζουμε την κατοπτρικότητα του αντικειμένου και πως θα εμφανίζεται.

Η καρτέλα Shading έχει τις ιδιότητες της επισκίασης του αντικειμένου.

Η Game Settings ρυθμίζει τον τρόπο που θα εμφανίζεται το αντικείμενο κατα την διάρκεια του παιχνιδιού.

Η Physics ρυθμίζει τις λειτουργίες του αντικειμένου, δίνοντάς του τη δυνατότητα να συμπεριφέρεται σαν ένα αντικείμενο στον πραγματικό κόσμο.

Η καρτέλα Transparency δίνει την ιδιότητα της διαφάνειας στο αντικείμενο, τα πιο διαδεδομένα στη χρήση τους ειναι τα Z Transparency και Raytrace. Η τιμή του Alpha απο κάτω δείχνει πόσο διαφανές ειναι το αντικείμενο.

Σε κάθε αντικείμενο μπορούμε να βάλουμε όσα Material θέλουμε. Μπορούμε να χρησιμοποιούμε όποιο μας βολεύει κάθε φορά, ή να χρησιμοποιήσουμε πάνω απο ένα ταυτόχρονα.

Για να το κάνουμε αυτό, γυρίζουμε το αντικείμενο σε Edit Mode, τσεκάρουμε τα σημεία στα οποία θέλουμε ενα συγκεκριμένο material και με το κουμπί Assign το εφαρμόζουμε. Σε Edit Mode

ξανα δημιουργούμε ένα νέο material και το εφαρμόζουμε και αυτό με τον ίδιο τρόπο σε κάποια άλλα μέρη του ίδιου αντικειμένου.

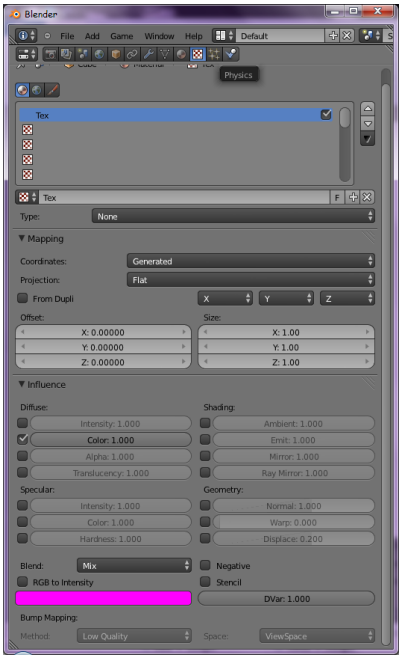
C.4.2 Textures

Στην Εικόνα 79 έχουμε την καρτέλα Texture. Και πάλι μία λίστα εμφανίζεται την οποία μπορούμε να διαχειριστούμε, όπως θέλουμε, καθώς και να αλλάξουμε το όνομα κάθε texture που δημιουργούμε. Απο τη λίστα μπορούμε να επιλέξουμε ποιά textures θέλουμε να φαίνονται.

Στο πεδίο Type έχουμε διάφορα είδη υφών, όπως τα Clouds, Noise, Marble, Ocean, Wood και διάφορα άλλα, τα οποία χρησιμοποιούνται ανάλογα με το τι υφή θέλουμε να έχει ένα αντικείμενο. Το πεδίο του Type με το οποίο ασχολούμαστε περισσότερο ειναι το Image or Movie.

Στο πεδίο Influence στην επιλογή Blend μπορούμε να διαλέξουμε με ποιό τρόπο θα ανακατεύονται τα textures μεταξύ τους, αν είναι παραπάνω απο ένα.

Ανοίγουμε μία εικόνα της αρεσκείας μας και αμέσως εμφανίζεται πάνω στο αντικείμενο. Αν η εικόνα όμως δεν εφαρμόζει τέλεια πάνω σε αυτό σύμφωνα με τη γεωμετρία του, αν ειναι για παράδειγμα ένα σύνθετο αντικείμενο, τότε χρησιμοποιούμε μια διαδικασία που λέγεται Mapping.



Εικόνα 79 - Texture

Sand Plane

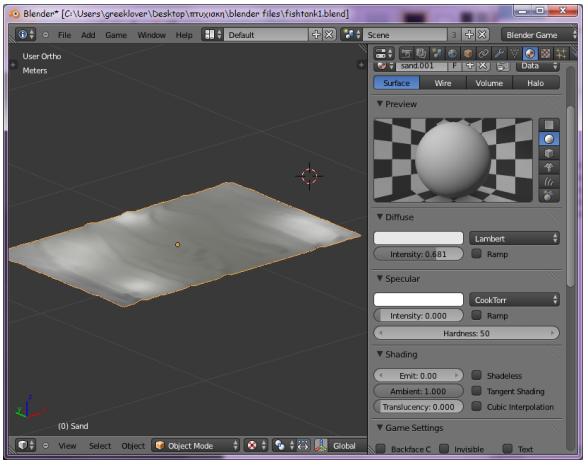
Όταν έχουμε ένα απλό αντικείμενο, όπως είναι το Plane της άμμου για παράδειγμα, η διαδικασία που ακολουθούμε για να του βάλουμε texture είναι η εξής.

Δημιουργούμε ένα νέο material για το Plane, του δίνουμε ένα όνομα, μηδενίζουμε το Intensity στο Specular για να μην ειναι αστραφτερό και απενεργοποιούμε το Backface Calling στο Game Settings ώστε να φαίνεται ολόκληρο το αντικείμενο κατά το τρέξιμο του παιχνιδιού.

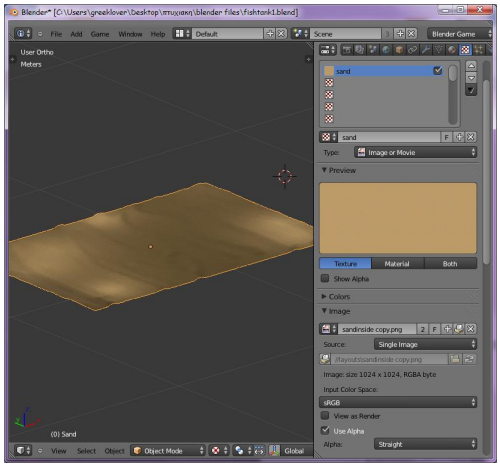
Στη συνέχεια δημιουργούμε ένα νέο texture, του δίνουμε ένα όνομα, και διαλέγουμε το Image or Movie. Ανοίγουμε μία εικόνα έτοιμη ή επεξεργασμένη σε κάποιο πρόγραμμα τύπου Photoshop και αυτή αυτόματα τοποθετείται πάνω στο αντικείμενο.

Το αποτέλεσμα για την άμμο είναι αυτό της Εικόνας 81.

Τέλος πατάμε την επιλογή Pack an Image.



Εικόνα 80 – διαμόρφωση του Plane της άμμου



Εικόνα 81 – προσθήκη Texture πάνω στο Plane

C.4.3 UV Mapping / Χαρτογράφηση

Water Plane

Στο plane του νερού που φτιάξαμε, επιλέγουμε στα Materials το Z-Transparency:0,104, δηλαδή το πόσο διαφανές θα είναι το αντικείμενο και του δώσαμε τιμή.

Για να κάνουμε την επιφάνεια του Water να φαίνεται ρεαλιστική, θα τοποθετήσουμε σαν texture στο Plane μία εικόνα, η οποία ειναι φτιαγμένη απο πολλες μικρότερες εικόνες, έτσι ώστε το αποτέλεσμα να είναι όσο πιο ρεαλιστικό γίνεται. Την εικόνα αυτήν θα τη χρησιμοποιήσουμε σαν animation πάνω στο Plane για να έχουμε την ψευδαίσθηση της κίνησης του νερού.

Όπως φαίνεται και στην Εικόνα 82, ανοίγουμε παράλληλα με το 3D View και έναν UV/Image Editor.

Στον UV/Image Editor ανοίγουμε την εικόνα που θα βάλουμε σαν texture. Επιλέγουμε το Plane και και σε Edit Mode το κάνουμε Unwrap. Απο το Texture του Properties Editor, διαλέγουμε σαν τύπο το Image or Movie, και ανοίγουμε την ίδια εικόνα. Στο Mapping διαλέγουμε στο Coordinates το UV. Τώρα το texture φαίνεται πάνω στο Plane.

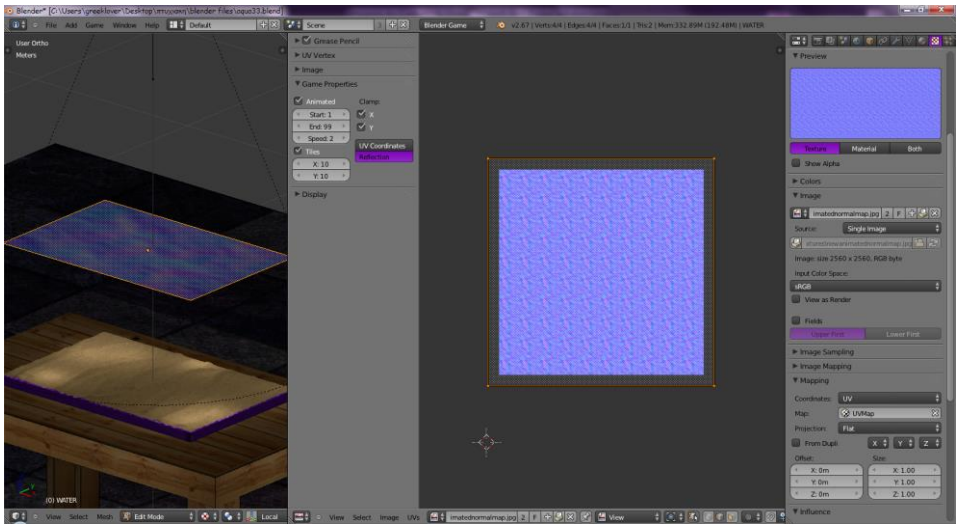
Στο UV/Image Editor, επιλέγουμε το UV Layout του Plane, με το **S** μπορούμε να το μεγαλώσουμε και να το μικρύνουμε. Κάνοντας το αυτό, ορίζουμε το πώς θα φαίνεται πάνω στο Plane το texture. Όσο πιο κοντά είναι το μέγεθος του UV Layout στο μέγεθος του texture, τόσο πιο λεπτομερές φαίνεται το texture στο Plane, ενώ όσο το μικραίνουμε ή μεγαλώνουμε τόσο χαλάει η ανάλυσή του.

Για να φτιάξουμε το animation, ανοίγουμε την καρτέλα Properties του UV/Image Editor, πατώντας το **N**. Όπως φαίνεται, στην Εικόνα 68 στο Game Properties τσεκάρουμε το Animated και ορίζουμε το Start:1, End:99, Speed:2,τσεκάρουμε τα Clamp X,Y και τα Tiles στα οποία δίνουμε την τιμή 10.

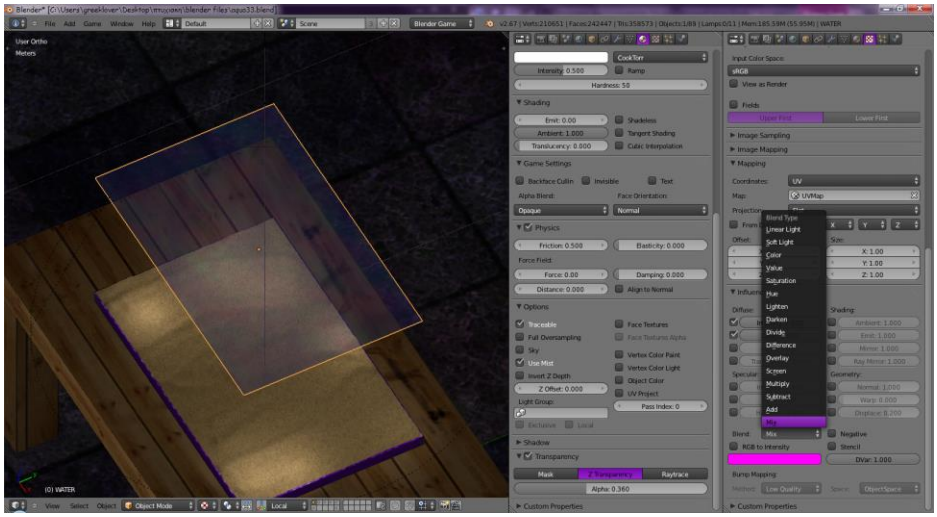
Τα Tiles χρησιμοποιούνται για να επαναλαμβάνεται ένα texture παραπάνω απο μία φορές σε ένα object. Με αυτόν τον τρόπο μπορούμε να δημιουργήσουμε κινούμενα textures.

Ανάλογα με τα Tiles που έχουμε ορίσει, ρυθμίζουμε τις επιλογές του Animated, από ποιο, μέχρι ποιό frame θα τρέξει και πόση θα είναι η ταχύτητά του.

Τέλος τα Clamp X,Y τα τσεκάρουμε, ώστε να μπορούμε να μεγαλώσουμε το UV Layout, χωρίς να τετραγωνίζεται η εικόνα.



Εικόνα 82 – ρυθμίσεις για τη δημιουργία Texture Animated



Εικόνα 83 – εφαρμογή του Transparency

Τέλος ενεργοποιούμε το Transparency στην καρτέλα Materials και μπορούμε να αλλάξουμε τα χρώματα του texture όπως εμφανίζονται πάνω στο plane, από την καρτέλα **Influence→Blend**, έχουμε διαλέξει το Mix.

Όταν πατήσουμε το P και ξεκινήσει να τρέχει το παιχνίδι, παρατηρούμε ότι η επιφάνεια του Plane μοιάζει με την κίνηση του νερού.

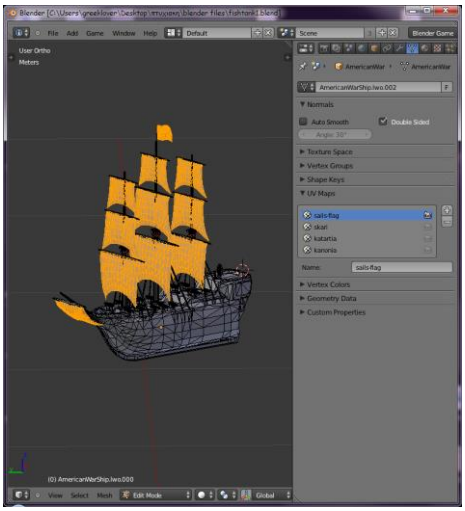
Ship Mesh

Για πιό σύνθετα αντικείμενα, όπως είναι το καράβι, η διαδικασία είναι πιό πολύπλοκη. Στο καράβι θα πρέπει να βάλουμε άλλο texture για το ξύλινο μέρος του, άλλο για τα πανιά και τη σημαία και άλλο για τα κανόνια του. Επείδή όμως και αυτά μεμονωμένα, δεν είναι επίπεδες επιφάνειες, αλλά κάποια είναι καμπυλωτά ή κυλινδρικά, για μεγαλύτερη ακρίβεια θα χρησιμοποιήσουμε και εδώ τη μέθοδο **UV Mapping**.

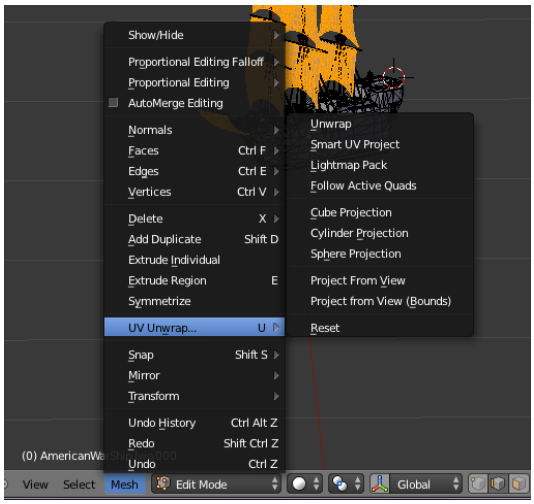
Οι υφές χρειάζονται συντεταγμένες χαρτογράφησης για να καθορίσουν πως θα εφαρμοστούν γεωμετρικά. Η χαρτογράφηση προσδιορίζει πως θα τυλιχθεί η υφή γύρω από το αντικείμενο. Λειτουργεί με το να θέτει ένα ζεύγος από συντεταγμένες για να οδηγούν τη διαδικασία χαρτογράφησης. Αυτές οι συντεταγμένες μπορούν να προέλθουν από παντού, συνήθως από το αντικείμενο πάνω στο οποίο θα εφαρμοστούν.

Η UV χαρτογράφηση είναι ένας ακριβής τρόπος για να χαρτογραφειθεί 2D περιεχόμενο υφής σε 3D επιφάνεια. Κάθε κορυφή του πλέγματος έχει τις δικές της UV συντεταγμένες οι οποίες μπορούν να "λυθούν" και να απλωθούν σαν δέρμα. Θα μπορούσαμε να φανταστούμε τις συντεταγμένες σαν μια χαρτογράφηση που λειτουργεί σε δισδιάστατο επίπεδο.

Αρχικά πηγαίνουμε στην καρτέλα Object Data του Properties Editor. Στην επιλογή UV Maps δημιουργούμε ενα νέο απο το + και του δίνουμε ένα όνομα. Στη συνέχεια επιλέγουμε το αντικείμενο και μεταφερόμαστε σε Edit Mode, όπου επιλέγουμε ποιά κομμάτια του αντικειμένου θέλουμε να συμπεριλάβουμε στο UV Map που φτιάξαμε. Για το καράβι φτίαξαμε 4 UV Maps, ένα για τα πανιά και τη σημαία, ένα για το σκαρί του πλοίου, ένα για τα κατάρτια και ένα για τα κανόνια. Μεταφερόμαστε σε Edit Mode και για κάθε UV Map επιλέγουμε τις αντίστοιχες περιοχές με τα εργαλεία Circle Select (**C**) και Border Select (**B**).



Εικόνα 84 – επιλογή με το C συγκεκριμένων σημείων στο object



Εικόνα 85 – UV Unwrap

Αφού επιλέξουμε τις περιοχές που θέλουμε να συμπεριλάβουμε σε κάποιο UV Map, επιλέγουμε απο το μενού του 3D View Editor **το Mesh→UV Unwrap** ή πατάμε το **U**.
Μας εμφανίζονται διάφορα είδη ξεδιπλώματος: [64]

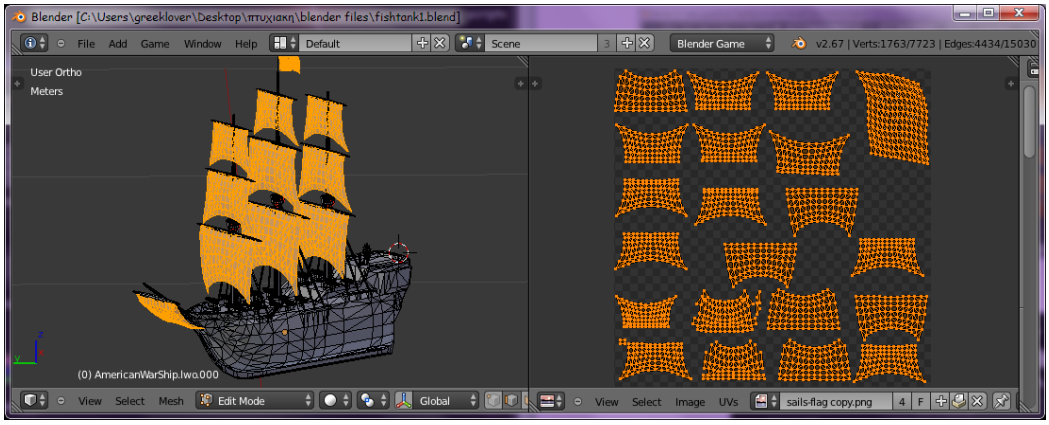
- Unwrap: ξεδιπλώνει ένα αντικείμενο ανάλογα με τις πλευρές του
- Smart UV Project: για πιο σύνθετα αντικείμενα τα κάνει δισδιάστατα με βάση ένα σημείο ξεδιπλώματος που έχουμε ορίσει εμείς
- Lightmap Pack και Follow Active Quads: επισημάνουμε στο αντικείμενο, ποιές ακμές θέλουμε να είναι οδηγός στο ξεδίπλωμα του
- Cube Projection: προβολή του κύβου σε δισδιάστατο περιβάλλον
- Cylinder Projection: προβολή του κυλίνδρου
- Sphere Projection: προβολή σφαίρας
- Project From View και Project From View (Bounds): προβολή του αντικειμένου όπως το βλέπουμε εκείνη τη στιγμή στο τρισδιάστατο επίπεδο

Διαλέγουμε το Smart UV Project.

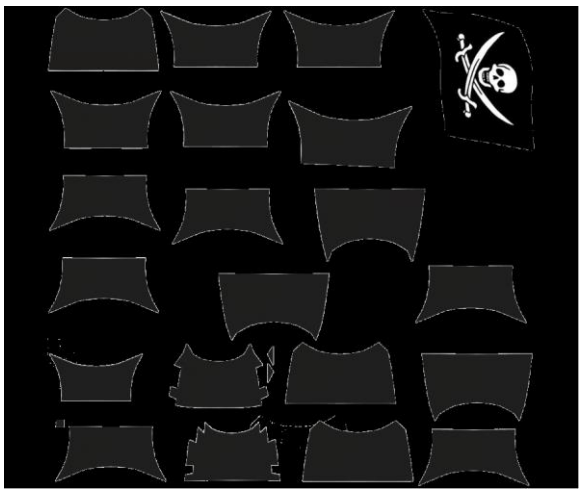
Χωρίζουμε το 3D View Editor στη μέση και ανοίγουμε ένα UV/Image Editor στον οποίο θα εμφανιστεί το αποτέλεσμα του ξεδιπλώματος σαν μια δισδιάστατη εικόνα.

Απο το μενού του UV/Image Editor, πατάμε **UVs→Export UV Layout**. Έτσι εξάγουμε το layout με τη μορφή .png, ώστε να μπορούμε να το επεξεργαστούμε με ένα πρόγραμμα και να του βάλουμε πάνω κάποια εικόνα σαν texture.

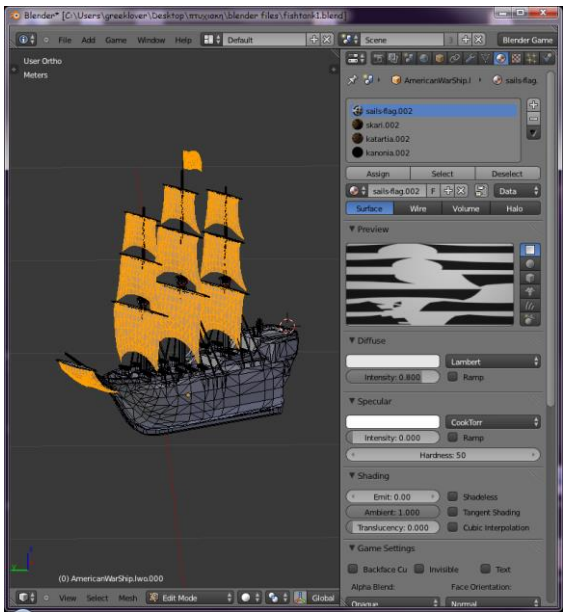
Η Εικόνα 87 είναι μια εικόνα που φτιάχτηκε σε Photoshop για να χρησιμοποιηθεί σαν texture του συγκεκριμένου layout.



Εικόνα 86 – αποτέλεσμα του Unwraping σε ένα UV/Image Editor



Εικόνα 87 – layout φτιαγμένο σε Photoshop

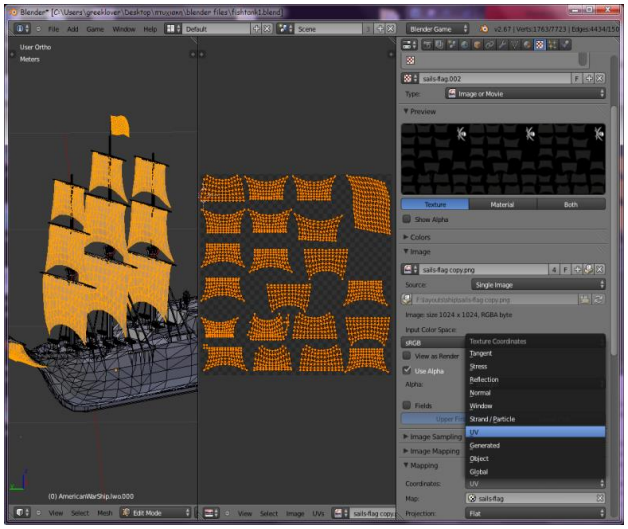


Εικόνα 88 – δημιουργία 4 Material

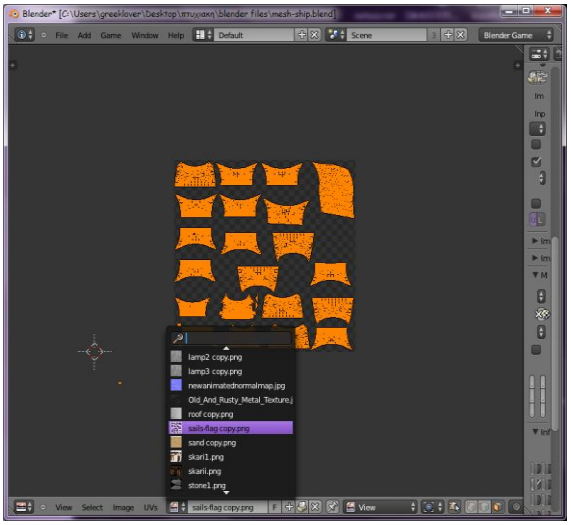
Στη συνέχεια στο Materials δημιουργούμε 4 material στα οποία δίνουμε τα ίδια ονόματα με τα UV Maps (εικόνα 88).

Για κάθε Material που δημιουργούμε, μπορούμε να δημιουργήσουμε τουλάχιστον ένα Texture. Μπορούμε να φτιάξουμε μια λίστα με Textures για κάθε Material και να τους δώσουμε ονόματα.

Επιλέγουμε το Image or Movie και ανοίγουμε την .png εικόνα που φτιάξαμε, στο παράδειγμα μας, απο την καρτέλα Mapping αλλάζουμε το Coordinates δηλαδή τις συντεταγμένες σε UV. Απο κάτω στο Map επιλέγουμε το UV Map στο οποίο ανήκουν τα αντικείμενα που επεξεργαζόμαστε.



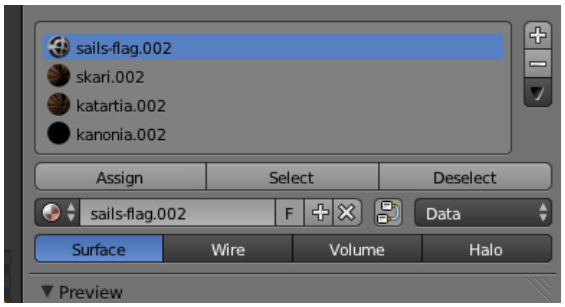
Εικόνα 89 – αλλαγή των συντεταγμένων σε UV



Εικόνα 90 – άνοιγμα του layout που χρησιμοποιήσαμε στον UV/Image Editor

Στο παράθυρο UV/Image Editor ανοίγουμε την εικόνα που φτιάξαμε, για να τη βλέπουμε και εκεί.

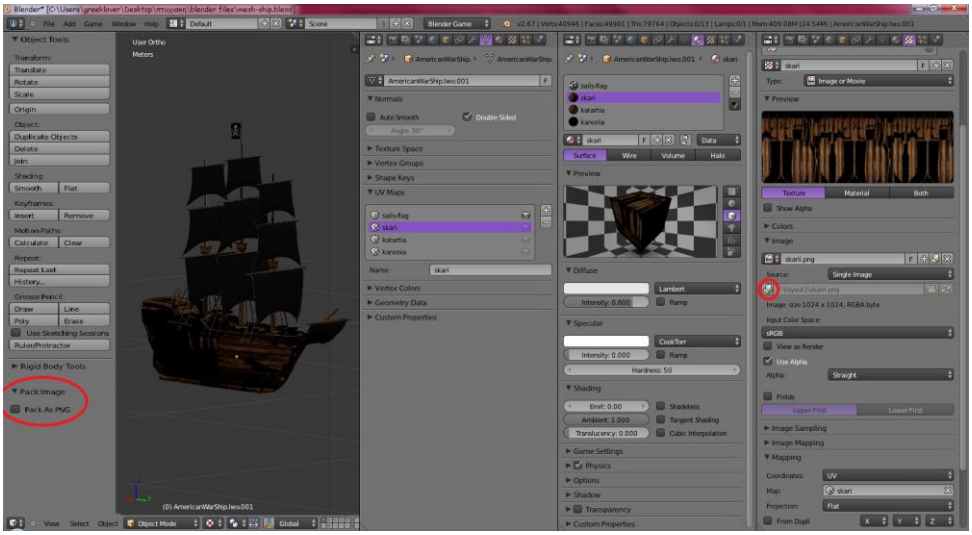
Την ίδια διαδικασία ακολουθούμε και για τα υπόλοιπα UV Maps. Για κάθε ένα δημιουργούμε ένα material και ένα texture. Αφού τελειώσουμε κάποιο το τοποθετούμε πάνω στο αντικείμενο, πατώντας το κουμπι Assign, όλα αυτά πρέπει να γίνονται σε Edit Mode. Απο τα κουμπιά Select και Deselect, επιλέγονται πάνω στο αντικείμενο τα μέρη εκείνα που ανήκουν στο UV Map που είναι διαλεγμένο.



Εικόνα 91 – η καρτέλα material σε Edit Mode με την επιλογή Assign

Στην Εικόνα 92 είναι οι 4 Editors με τις ρυθμίσεις που κάναμε για τα εργαλεία Object Data, Material και Texture και το τελικό αποτέλεσμα.

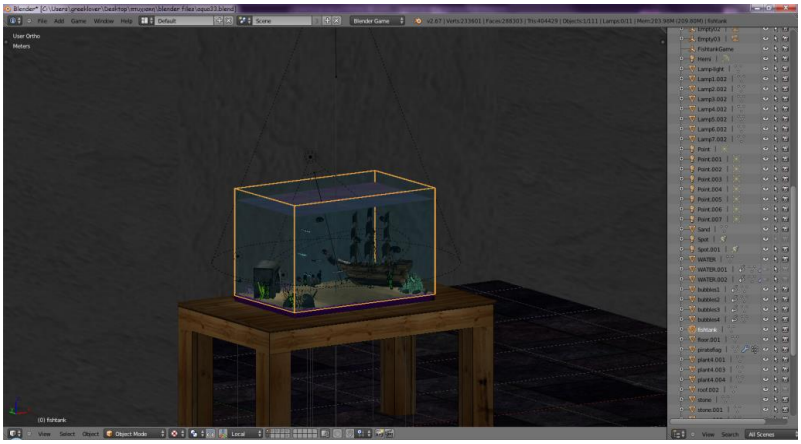
Τέλος στο κυκλωμένο σημείο στο παράθυρο Texture υπάρχει ενα κουμπί το οποίο σώζει σαν πακέτο το αντικείμενο ή το κομμάτι του αντικειμένου, μαζί με την εικόνα που τοποθετήσαμε σε αυτό, κατόπιν τσεκάρουμε το Pack As PNG. Με αυτή τη διαδικασία όταν θα ξεκινήσουμε να τρέχουμε το παιχνίδι, θα εμφανίστουν οι υφές των αντικειμένων.



Εικόνα 92 – οι καρτέλες του Properties Editor που χρησιμοποιούμε για πολλαπλά materials

C.4.4 Αποτέλεσμα

Το αποτέλεσμα που θα έχουμε αν εφαρμόσουμε τα παραπάνω, σε όλα τα αντικείμενα της σκηνής μας θα είναι αυτό της Εικόνας 93.



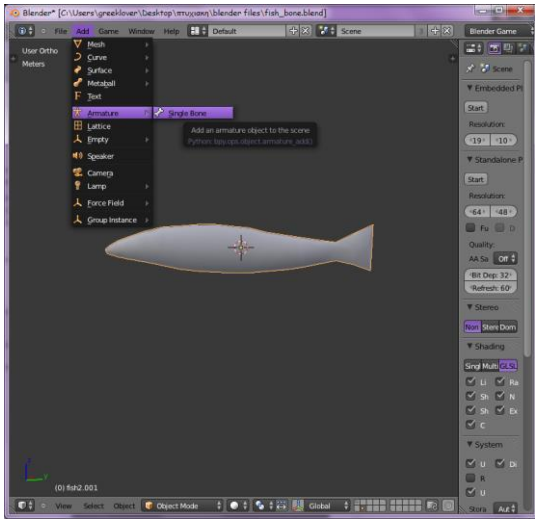
Εικόνα 93 – τελικό αποτέλεσμα μετά από τη χρήση των Materials και Textures

C.5 Rigging

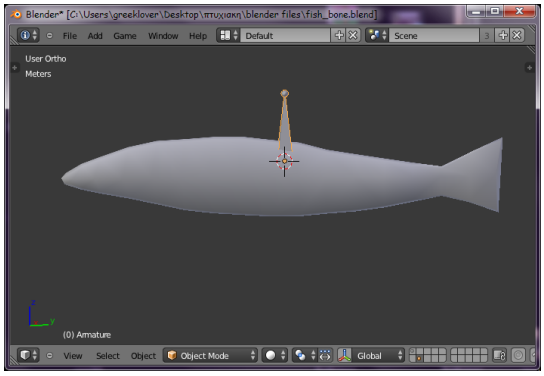
Όπως σε όλα τα 3D προγράμματα έτσι και στο blender μπορούμε να δώσουμε κινητικές ικανότητες σε ένα αντικείμενο. Η διαδικασία με την οποία το κάνουμε αυτό ονομάζεται **Rigging**. Στην ουσία επισυνάπτουμε έναν σκελετό στο αντικείμενο, έτσι ώστε να μπορούμε με αυτόν να του δίνουμε κινήσεις και γενικά να το παραμορφώνουμε.

Όπως φαίνεται στην Εικόνα 94 για να προσθέσουμε ένα Bone πηγαίνουμε στο μενού **Add→Armature→Single Bone**. Έτσι Στο σημείο που είναι τοποθετημένο το 3D Cursor θα εμφανιστεί το Bone.

Το αποτέλεσμα της εισαγωγής όπως στις παρακάτω εικόνες:



Εικόνα 94 – προσθήκη ενός νέου Bone

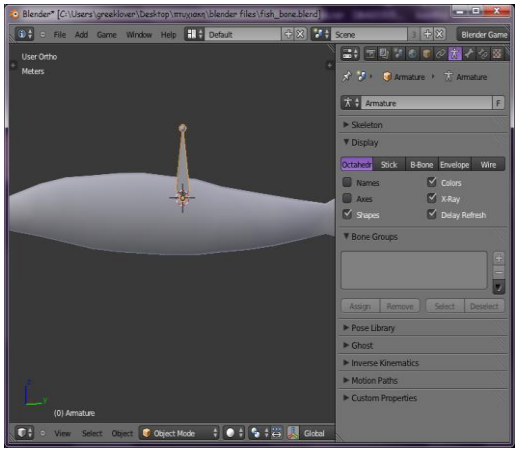


Εικόνα 95 – τοποθέτησή του στο object

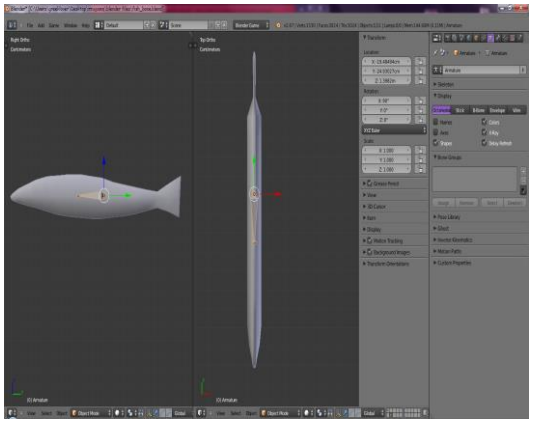
Με το Bone τσεκαρισμένο, ανοίγουμε το παράθυρο των Properties.

Αρχικά πηγαίνουμε στο tab Object Data, όπου ψηλά μας δείχνει το όνομα του αντικειμένου που προσθέσαμε και το οποίο μπορούμε εμείς να αλλάξουμε από κάτω. Στην καρτέλα Display μπορούμε να επιλέξουμε την μορφή που θέλουμε να έχουν τα Bones. Αρχικά χρησιμοποιούμε την Octahedral μορφή, η οποία είναι η πιο εύχρηστη για επεξεργασία. Όταν ολοκληρώσουμε το Armature μπορούμε να επιλέξουμε τη Wire μορφή, έτσι ώστε να είναι πιο διακριτικό το Bone πάνω στο αντικείμενο.

Στα checkboxes από κάτω τσεκάρουμε τι θέλουμε να φαίνεται την ώρα που δουλεύουμε με τα Bones. Επιλέγουμε το X-Ray ώστε να βλέπουμε μέσα από το αντικείμενο το Bone και να μπορούμε πιο εύκολα να το πειράξουμε.



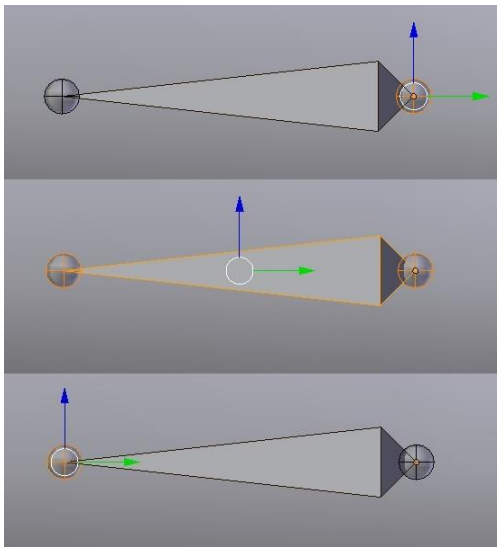
Εικόνα 96 - Object Data του Bone



Εικόνα 97 – επεξεργασία του bone σε 2 όψεις του 3d View

Κάνουμε Split το παράθυρο 3d View έτσι ώστε να βλέπουμε δύο πλευρές του αντικειμένου, όπως φαίνεται στην Εικόνα 97 έχουμε στο ένα παράθυρο την όψη Right Ortho (hotkeys: **numpad3** & **numpad5**) και στην δίπλα του την Top Ortho (**numpad7** & **numpad5**). Βλέποντας έτσι δύο όψεις, φέρνουμε το Bone στη θέση που θέλουμε και το περιστρέφουμε επίσης όπως μας εξυπηρετεί. Απο το Tab Properties (**N**) στην καρτέλα Transform βλέπουμε πόσο το έχουμε μετακινήσει ή περιστρέψει.

Για να συνεχίσουμε την δημιουργία του σκελετού, μεταφερόμαστε σε Edit Mode με το Bone επιλεγμένο. Στην παρακάτω εικόνα βλέπουμε ότι σε Edit Mode μπορούμε να επιλέξουμε το Bone σε τρία διαφορετικά σημεία.



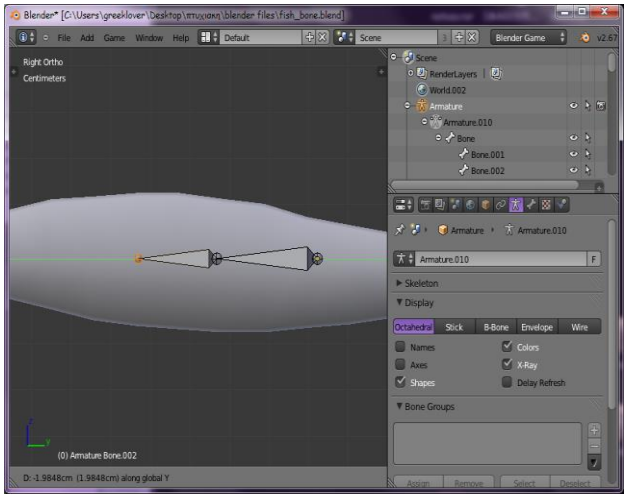
Εικόνα 98 – σημεία που μπορούμε να επιλέξουμε το bone σε Edit Mode

Στο πρώτο σημείο, διαλέγουμε το Bone από το Root ή Head του Bone. Απο τη βάση του δηλαδή. Τη βάση μπορούμε μόνο να τη μετακινήσουμε, αλλάζοντας έτσι το σημείο από το οποίο ξεκινάει το Bone χωρίς όμως αυτο να επηρεάζει την κατάληξή του (Tail).

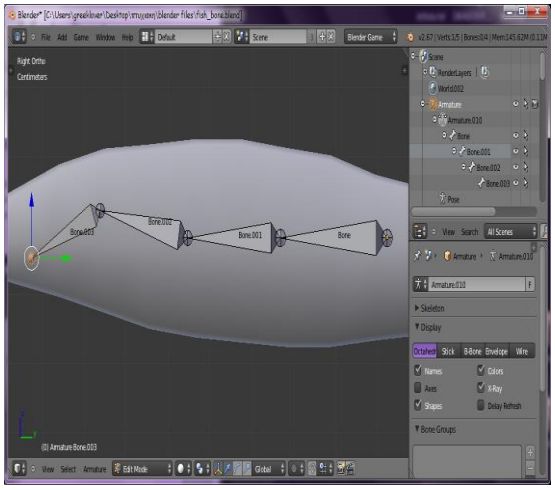
Στο δεύτερο σημείο, διαλέγουμε το Body από το Bone, διαλέγουμε δηλαδή ολόκληρο το Bone και έτσι οτι αλλαγή κάνουμε σε αυτό θα γίνει και σε ολόκληρο το αντικείμενο.

Στο τρίτο σημείο, διαλέγουμε το Tail, την κατάληξη του Bone. Μπορούμε να το μετακινήσουμε αλλάζοντας το που τελειώνει το Bone χωρίς όμως να επηρεάζεται το Root του. Επίσης από το Tail μπορούμε να περιστρέψουμε το Bone γύρω από τον άξονα μεταξύ Head και Tail.

Για να επεκτείνουμε τον σκελετό μας, πρέπει να προσθέσουμε ένα καινούργιο Bone. Αυτό δεν θα γίνει από το **Add→Armature→Bone** ξανά, γιατί έτσι θα δημιουργήσουμε ένα νέο Armature. Εμείς θέλουμε να φτιάξουμε ένα νέο Bone το οποίο να είναι συνδεδεμένο από το πρώτο και να εξαρτάται από αυτό, όπως λειτουργεί και ένας σκελετός. Για να το πετύχουμε αυτό, έχοντας επιλεγμένο το Tail του πρώτου Bone πατάμε στο μενού **Armature→Extrude** ή πατάμε το **E** και με το ποντίκι δίνουμε την κατεύθυνση που θέλουμε να έχει το νέο Bone. Για περισσότερη ακρίβεια μπορούμε να επιλέξουμε και τον άξονα στον οποίο θέλουμε να επεκταθεί το νέο Bone, πατώντας π.χ. **E** και **y** (Εικόνα 99)



Εικόνα 99 – δημιουργία νέου bone πάνω στον y άξονα



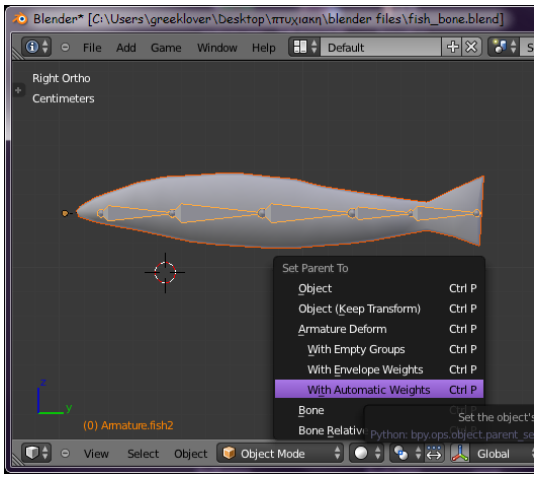
Εικόνα 100 – ονόματα των bones

Με τον ίδιο τρόπο μπορούμε να προσθέσουμε και άλλα Bones προς όποια κατεύθυνση θέλουμε. Κάθε Bone που προστίθεται, εξαρτάται από το προηγούμενο του. Στην Εικόνα 86 φαίνονται

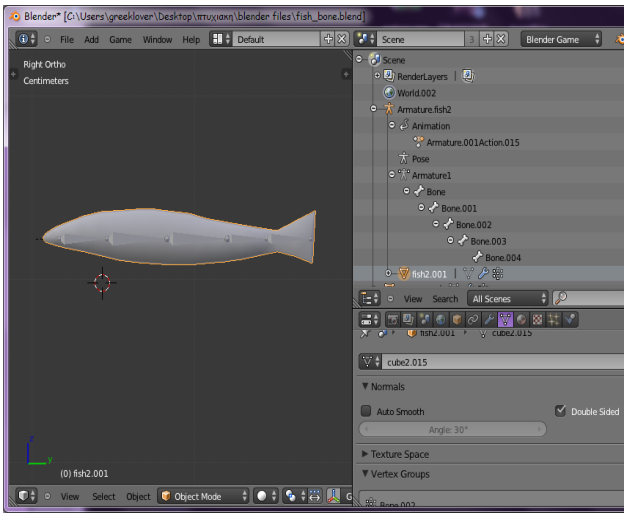
τα Bones που δημιουργήθηκαν με τα ονόματα τους, και στο παράθυρο Outliner το διάγραμμα με τις σχέσεις μεταξύ των Bones.

Ένας άλλος τρόπος να φτιάξουμε πολλά Bones τα οποία να θέλουμε να είναι ευθυγραμμισμένα σε ένα άξονα είναι αφού βάλουμε το πρώτο Bone να το μεγαλώσουμε όσο θέλουμε να είναι ο σκελετός μας και στη συνέχεια να πατήσουμε **Armature→Subdivide** (**W**) και να χωρίσει στα δύο το Bone. Όσες φορές το πατάμε τόσες φορές υποδιαιρεί το Bone.

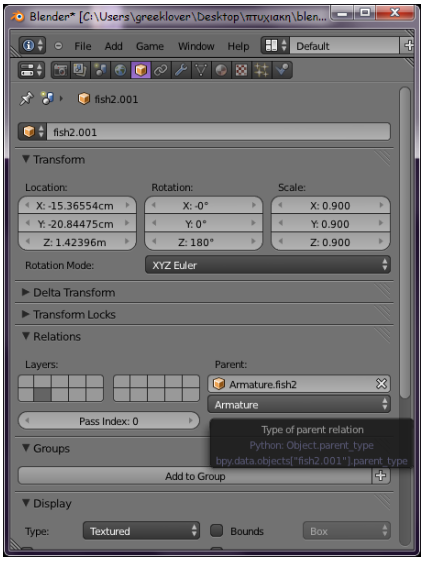
Αφού τελειώσουμε τον σκελετό, μπορούμε να ξεκινήσουμε να του δίνουμε κινήσεις. Πριν αυτό όμως, πρέπει να τον ενώσουμε με το αντικείμενο, έτσι ώστε να επιδρά σε αυτό όταν κινείται. Θα κάνουμε τον σκελετό parent του object. Επιλέγουμε πρώτα το αντικείμενο και με το shift πατημένο επιλέγουμε και τον σκελετό. Πάμε στο **Object→Parent→Armature Deform With Automatic Weights** ή εναλλακτικά πατάμε **Ctrl-P→ Armature Deform With Automatic Weights** (Εικόνα 101). Με αυτόν τον τρόπο κάναμε το Armature parent του αντικειμένου με κάθε Bone να επηρεάζει αυτόματα την περιοχή του Object στην οποία βρίσκεται. Στην Εικόνα 102, βλέπουμε στο Outliner ότι τώρα μέσα στο Armature βρίσκεται και το αντικείμενο, το οποίο είναι το fish. Στο παράθυρο Properties με επιλεγμένο το αντικείμενο πηγαίνουμε στο Tab Objects και στην καρτέλα Relations βλέπουμε ποιό είναι το Parent του αντικείμενου. Στην περίπτωση μας που το Parent είναι Armature πρέπει να αλλάξουμε το Parent σε Armature όπως στην Εικόνα 103.



Εικόνα 101 – επιλογή του είδους του parenting που θέλουμε



Εικόνα 102 – parenting των bones του armature



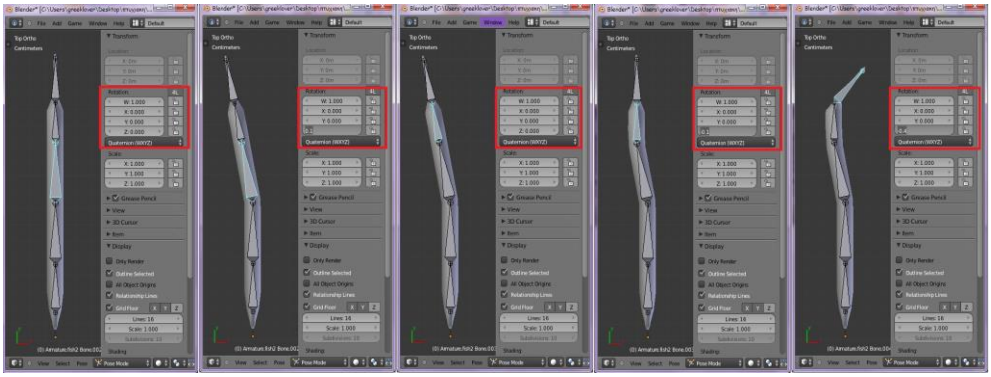
Εικόνα 103 – αλλαγή του Parent σε Armature

Για να φτιάξουμε την κίνηση του σκελετού και άρα του αντικείμενου, πρέπει έχοντας επιλεγμένο το σκελετό να αλλάξουμε από Edit mode σε Pose Mode. Σε Pose Mode μπορούμε να

κινήσουμε το σκελετό ή να τον περιστρέψουμε, ώστε να δώσουμε την αίσθηση της κίνησης, χωρίς όμως να αλλάξουμε την θέση που του δώσαμε σε Object Mode.

Στο συγκεκριμένο παράδειγμα, η κίνηση των Bones είναι περιστροφική στον z άξονα. Γι' αυτό τον λόγο θα αλλάξουμε το View σε Top Ortho (numpad7).

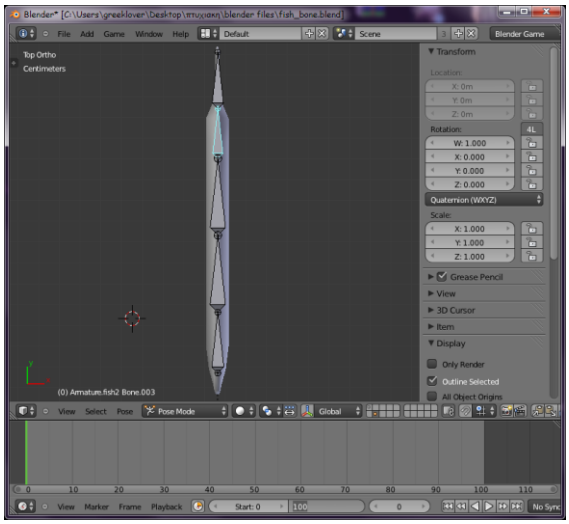
Επιλέγουμε το Bone που θέλουμε να περιστρέψουμε. Για μεγαλύτερη ακρίβεια τις περιστροφές τις κάναμε από το Properties **(N)→Transform→Rotation**. Βλέπουμε ότι όπως περιστρέφεται το κάθε Bone, περιστρέφεται και το αντίστοιχο τμήμα του αντικειμένου που είναι γύρω του. Επίσης παρατηρούμε ότι το επόμενο Bone του οποίου είναι parent ακολουθεί την ίδια κλήση.



Εικόνα 104 – δοκιμές κίνησης των bones

Για να καταγράψουμε την κίνηση στο αντικείμενο έτσι ώστε να κινείται όποτε θέλουμε, θα το κάνουμε Animation. Αφού αποφασίσουμε πως θα είναι οι κινήσεις των Bones και με ποια σειρά θα γίνονται, θα ακολουθήσει η καταγραφή τους.

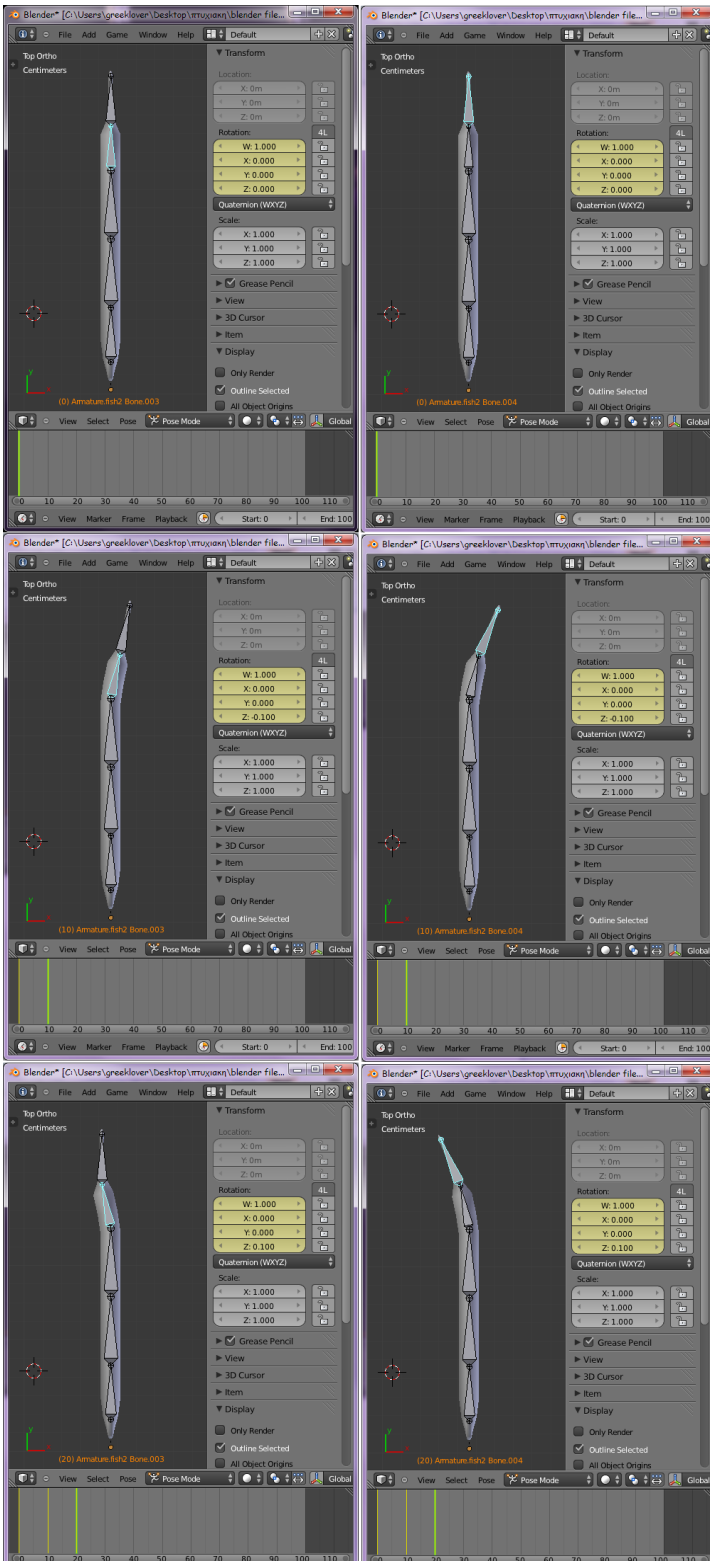
Κάνουμε Split το παράθυρο του 3d View και θα ανοίξουμε ένα Timeline. Ρυθμίζουμε πόσα frames θέλουμε να έχουμε.



Εικόνα 105 – editors για την διαδικασία καταγραφής των frames

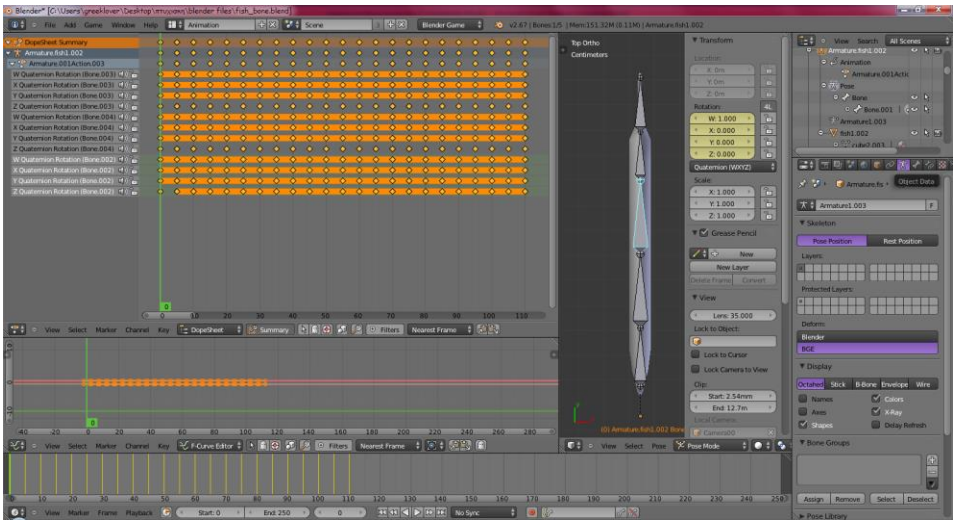
Για κάθε Bone που θέλουμε να περιστρέψουμε, θα βάλουμε keyframes σε κάθε frame στο οποίο θέλουμε να έχουμε αλλαγή κίνησης.

Στο frame 0 θα βάλουμε ένα keyframe με την αρχική κατάσταση των Bones. Στο πεδίο Rotation του properties θα κάνουμε δεξί κλικ και **Insert Keyframes**. Παρατηρούμε ότι τα πεδία με τις τιμές του Rotation καθώς και το frame 0 του timeline έγιναν κίτρινα, αυτό σημαίνει ότι το Keyframe τοποθετήθηκε για το συγκεκριμένο Bone με τις συγκεκριμένες τιμές. Στην Εικόνα 106 φαίνονται τα keyframes που έχουν μπει για κάθε Bone και οι τιμές τους.



Εικόνα 106 – τιμές των keyframes για κάθε bone

Για πιο λεπτομερή ανάλυση του animation, αλλάζουμε το Screen Layout από 3D View σε Animation.

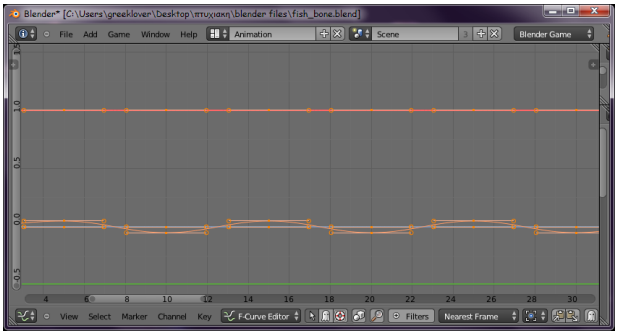


Εικόνα 107 – Animation View

Τα δύο νέα παράθυρα που βλέπουμε είναι το DopeSheet Editor και το F Curve Editor.

Στο το DopeSheet Editor έχουμε μία λίστα με τις ενέργειες που έγιναν σε κάθε Bone. Για κάθε Bone που επιλέγουμε σε Pose Mode επιλέγονται και οι αντίστοιχες ενέργειες στη λίστα. Οι ενέργειες αυτές είναι η περιστροφές που έγιναν σε κάθε άξονα για κάθε Bone. Δίπλα από κάθε ενέργεια εμφανίζονται τα keyframes που έχουμε βάλει. Όσα είναι ενωμένα με την πορτοκαλί γραμμή, δεν αλλάζει η τιμή τους κατά τη διάρκεια του animation. Επίσης σε αυτόν τον Editor μπορούμε να μετακινήσουμε πατώντας το G τα keyframes, επιλέγοντας με δεξί κλικ όσα θέλουμε ή με A όλα, μπορούμε να τα κάνουμε αντιγραφή – επικόλληση, να τα διαγράψουμε.

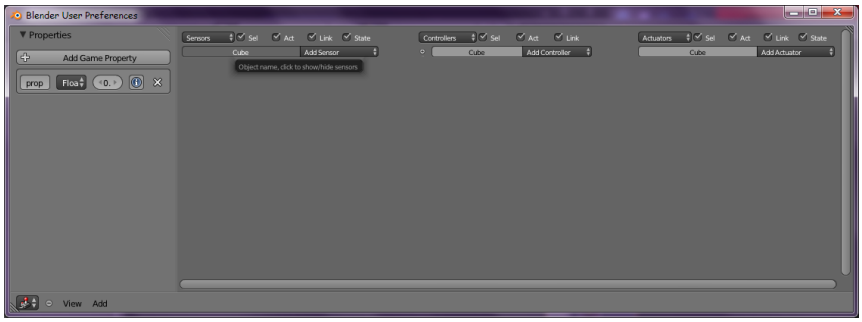
Στον F Curve Editor (Εικόνα 108) εμφανίζεται ένα διάγραμμα ταχύτητας-απόστασης το οποίο καθορίζει την ταχύτητα και την επιτάχυνση του αντικείμενου. Τα σημεία που φαίνονται σε αυτό είναι τα keyframes. Για να αλλάξουμε την ταχύτητα και την επιτάχυνση του αντικειμένου, πειράζουμε την άσπρη καμπύλη του διαγράμματος. Για να σταθεροποιήσουμε την ταχύτητα μεταξύ δύο keyframes, τα επιλέγουμε και από το μενού πατάμε **Key→ Interpolation Mode→Linear** κάνει έτσι την γραμμή ευθεία, σβήνοντας την επιτάχυνση.



Εικόνα 108 - F Curve Editor

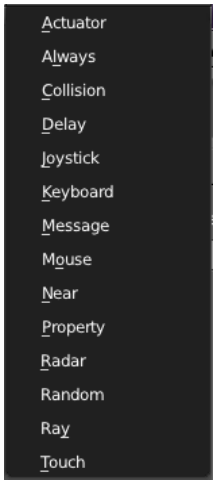
C.6 Game Logic

C.6.1 Game Logic Editor

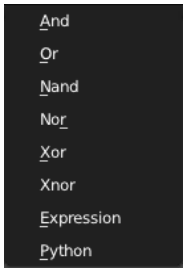


Εικόνα 109 - Game Logic Editor

Στην Εικόνα 109, βλέπουμε στα αριστερά ένα πεδίο με το όνομα Properties. Από εδώ μπορούμε να προσθέσουμε στο παιχνίδι ιδιότητες όπως για παράδειγμα το μέτρημα των πόντων και τη διάρκεια ζωής του παίχτη. Γράφουμε το όνομα της ιδιότητας, επιλέγουμε αν η μεταβλητή θα είναι μια ακέραια, πραγματική, ή λογική τιμή, ορίζουμε αν θα αρχίζει από το 0 ή από κάποιον άλλον αριθμό, και πατώντας το (i) επιτρέπουμε κατά το τρέξιμο του παιχνιδιού, να εμφανίζεται στην οθόνη το Property.



Εικόνα 110 - Sensors



Εικόνα 111 - Controllers

Εικόνα 112 - Actuators

C.6.2 Εφαρμογή Game Logic σε Objects

Για το παιχνίδι, θέλουμε να δώσουμε τις εξής εντολές στα αντικείμενά μας:

Το ψάρι να μετακινείται προς μία κυκλική κατεύθυνση και μόλις ακουμπάει τη γυάλα να στρίβει αλλού. Ταυτόχρονα να κινείται και το armature που του βάλαμε.

Τα φύκια να κουνάνε τον οπλισμό τους, σαν κίνηση.

Οι κάμερες να μπορούν να πηγαίνουν μπρος και πίσω και να στρίβουν δεξιά, αριστερά, πάνω και κάτω.

Το καμάκι εκτοξεύεται απο μία κάμερα ανάλογα με την φορά που έχει πάρει κάθε στιγμή.

Το αγκίστρι είναι ανεξάρτητο και μπορεί να κινηθεί δεξιά, αριστερά, πάνω, κάτω, καθώς και μπροστά και πίσω.

Σαν εφέ, έχουμε προσθέσει φυσαλίδες οι οποίες κατά τη διάρκεια του παιχνιδιού θα βγαίνουν από τα ψάρια. Τις φυσαλίδες τις φτιάξαμε με UV spheres τις οποίες σε Edit Mode, πολλαπλασιάσαμε και αλλάξαμε το μέγεθός τους. Ύστερα σε object mode, τους προσθέσαμε material όπου τους δώσαμε transparency. Τέλος όσο θα τρέχει το παιχνίδι, θέλουμε να εμφανίζεται στην οθόνη, ένας μετρητής με τους πόντους που μαζεύει ο παίχτης.

Για τον μετρητή, αλλά και για την κίνηση των καμερών, θα χρησιμοποιήσουμε αντικείμενα τύπου Empty.

a. Camera

Σε κάθε κάμερα θα τοποθετήσουμε ένα Empty. Επιλέγουμε την κάμερα και πατάμε **Shift S→Cursor To Selected**. Έτσι ο 3d Cursor θα τοποθετηθεί στη βάση της κάμερας. Στη συνέχεια από τον Info Editor πατάμε **Add→Empty→Plain Axis**. Βάλαμε έτσι ένα Empty στη βάση της κάμερας. Στη συνέχεια, επιλέγοντας πρώτα την κάμερα και μετά με Shift το Empty, πατάμε **Ctrl P→Set Parent To Object**, κάνοντας έτσι το Empty parent της κάμερας. Με αυτόν τον τρόπο ότι και να κάνουμε στο Empty θα έχει και επίδραση στην κάμερα.

Για κάθε Empty που βάζουμε σε κάθε κάμερα, πρέπει να του δίνουμε και ένα όνομα που να παραπέμπει στην κάμερα, έτσι ώστε να είναι πιο εύκολο στο χειρισμό του. Παράδειγμα για την Camera01 έχουμε το Empty01.

Camera01

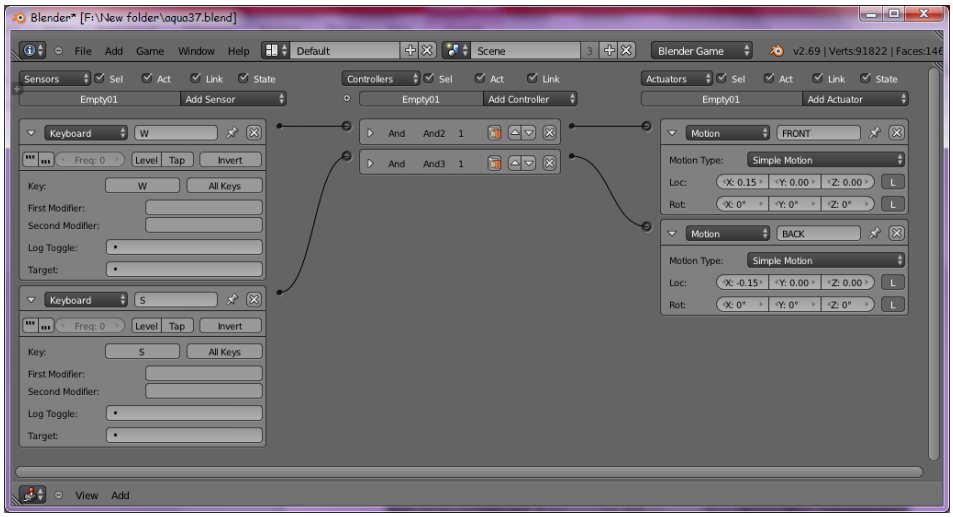
Για την κίνηση μπρος και πίσω του Empty και της κάμερας, χρησιμοποιήσαμε τις έτοιμες εντολές που έχει ο Game Logic Editor, ενώ για την περιστροφική κίνησή της, χρησιμοποιήσαμε ένα script σε Python με τον Text Editor.

Για την κίνηση μπροστά, θα χρησιμοποιήσουμε το πλήκτρο W. Στο πεδίο Sensors θα διαλέξουμε το Keyboard και στο πλαίσιο του Key, θα πατήσουμε το W. Στο πεδίο Controllers θα διαλέξουμε το And και στο Actuators το Motion. Στο Motion, θα επιλέξουμε το Simple Motion και την τιμή της μεταβολής της κίνησης θα την γράψουμε στο πεδίο Loc. Τα x, y, z είναι οι άξονες στους οποίους γράφουμε τη μεταβολή της κίνησης. Το L στο πλάι, το επιλέγουμε αν θέλουμε η κίνηση να γίνει στους local άξονες, αλλιώς θα γίνει στους global.

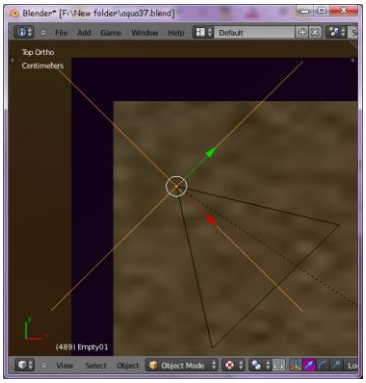
Όπως φαίνεται στην Εικόνα 113 βάλαμε να γίνεται κίνηση κατά 0.5 στον y άξονα του Empty και επιλέξαμε το L αφού και το Empty είναι ρυθμισμένο στους Local άξονες (Εικόνα 114)

Για την κίνηση προς τα πίσω, ανοίγουμε πάλι έναν Keyboard Sensor, έναν And Controller και έναν Motion Actuator. Διαλέγουμε το πλήκτρο να είναι το S και η κίνηση που θα κάνει να είναι Simple Motion στον y Local άξονα, με τιμή -0.5.

Τέλος ενώνουμε μεταξύ τους τα κουτιά.



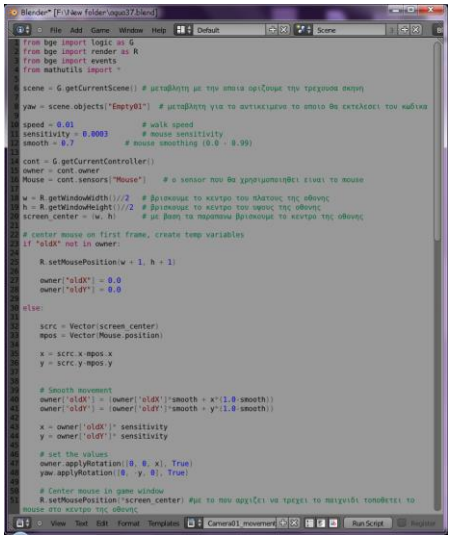
Εικόνα 113 - ρυθμίσεις για την κίνηση της Camera01 μέσω του Empty01



Εικόνα 114 – αλλαγή των αξόνων του Empty01 σε Local

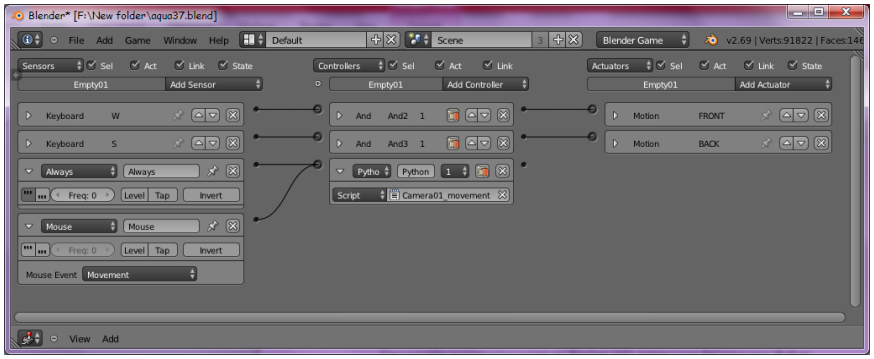
Πάμε στο 3D View και διαλέγουμε να βλέπουμε μέσα απο την όψη της Camera01. Αν πατήσουμε **P**, για να τρέξει το παιχνίδι, θα δούμε ότι με τα πλήκτρα **W** και **S**, θα προχωράμε μπρος και πίσω.

Για να φτιάξουμε την περιστροφική κίνηση που θα δίνει το ποντίκι θα χρησιμοποιήσουμε τον Text Editor στον οποίο μέσα θα βάλουμε το Script για την κίνηση του mouse (Εικόνα 115). Του δίνουμε όνομα "Camera01_movement"[82].

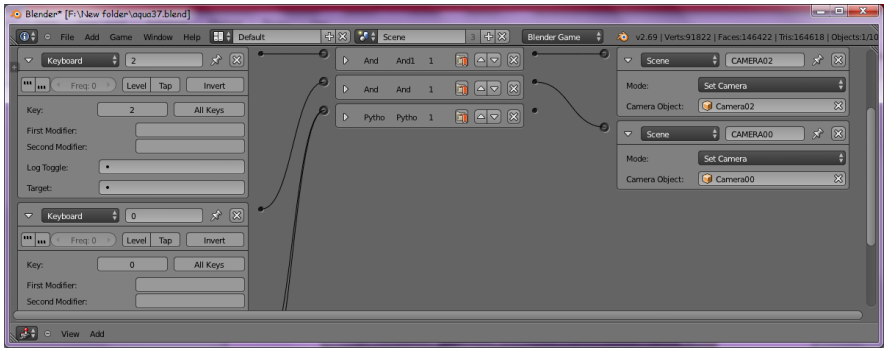


Εικόνα 115 – Python Script για την κίνηση της Camera01

Για να τρέξει το scirpt της Εικόνας 115, πρέπει να το βάλουμε στον Python Controller του Logic Editor, και θα το συνδέσουμε με δύο Sensors. Έναν Mouse με Mouse Event το Movement και έναν Always, όπως φαίνεται στην Εικόνα 116. Τον Mouse Sensor τον χρειαζόμαστε για να δείξουμε ότι οι εντολές του scirpt θα εκτελούνται απο το Mouse, ενώ τον Always Sensor για να εκτελούνται συνεχώς. Όταν χρησιμοποιούμε Python scirpt σαν Sensor, δεν χρειαζόμαστε, Actuator.



Εικόνα 116 - ρυθμίσεις για την κίνηση της Camera01 μέσω του Empty01



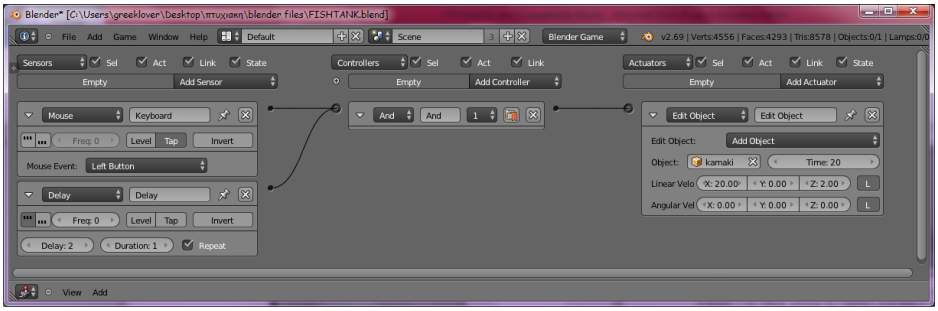
Εικόνα 117 – ρυθμίσεις για την κίνηση της Camera01 μέσω του Empty01

Στην Εικόνα 117 βλέπουμε άλλες δύο συνδέσεις που βάλαμε. Με το πάτημα του πλήκτρου 2 σαν Sensor, κάνουμε την Camera01 να αλλάζει με την Camera02. Στον Actuator χρησιμοποιήσαμε το Scene στο Mode βάλαμε το Set Camera και σαν κάμερα ορίσαμε την Camera02.

b. Αντικείμενο – Καμάκι

Το καμάκι θα εκτοξεύεται απο την Camera01, κάθε φορά που θα θέλουμε να πετύχουμε κάποιο ψάρι.

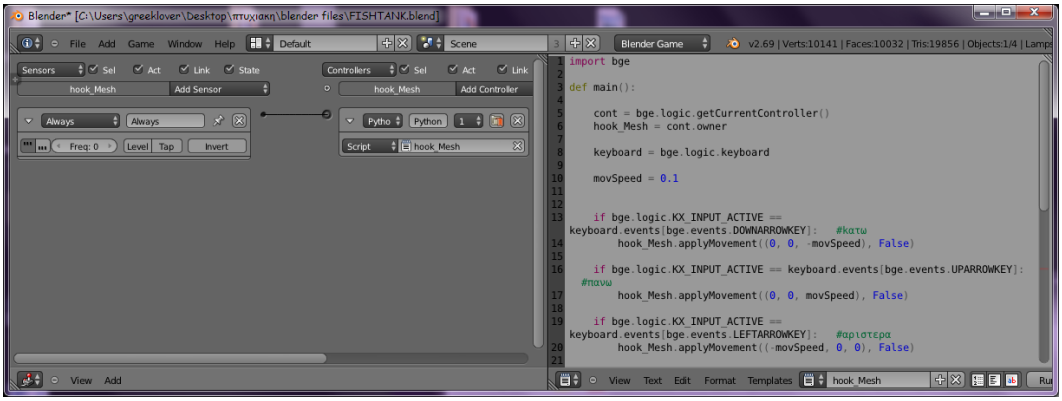
Για να το κάνουμε αυτό αρχικά τοποθετήσαμε το καμάκι σε ένα άλλο layer απο αυτό της κάμερας. Στην κάμερα προσθέσαμε ένα δεύτερο Empty στο οποίο στον Logic Editor βάλαμε σαν Sensors το Mouse με Mouse Event το Left Button και το Tap επιλεγμένο, και το Delay με ενεργοποιημένο σε true το Pulse Mode, με Delay:2 και Duration:1 και τσεκαρισμένο το Repeat. Τους δύο Sensors τους ενώσαμε μέσω ενός And Controller με τον Edit Object Actuator, του οποίου η ενέργεια θα είναι να προσθέτει ένα καμάκι στη σκηνή.



Εικόνα 118 – ρυθμίσεις για την κίνηση του αντικειμένου καμάκι

c. Αντικείμενο – Αγκίστρι

Το αγκίστρι φαίνεται απο την Camera02 και είναι ανεξάρτητο απο αυτήν. Με ένα Script[84] ορίζουμε την κίνηση που θα κάνει μέσα στο ενυδρίο.



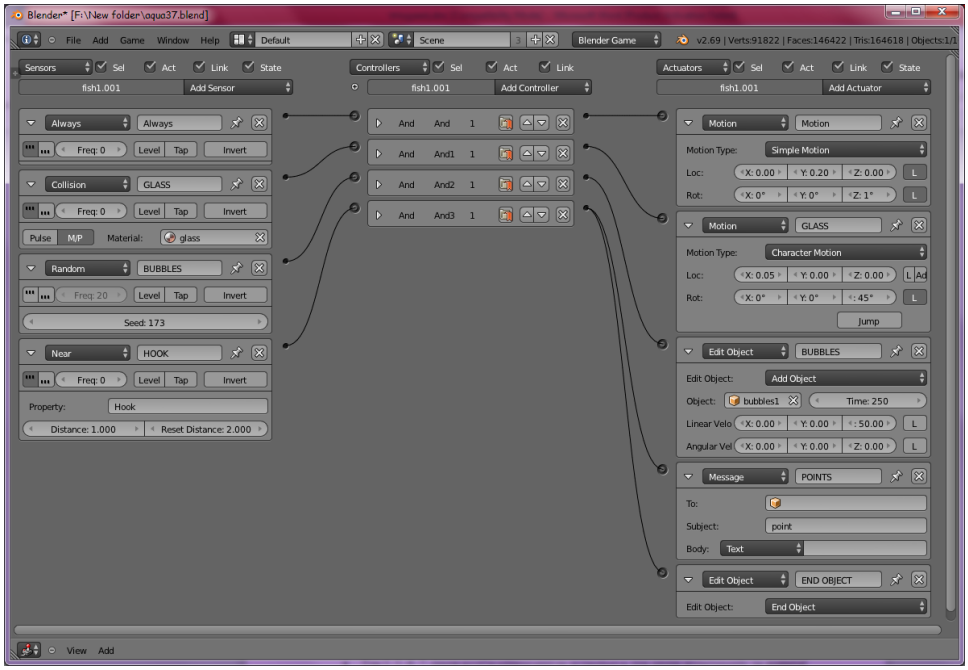
```
import bge

def main():

    cont = bge.logic.getCurrentController()
    hook_Mesh = cont.owner

    keyboard = bge.logic.keyboard

    movSpeed = 0.1


    if bge.logic.KX_INPUT_ACTIVE ==
keyboard.events[bge.events.DOWNARROWKEY]:   #κατω
        hook_Mesh.applyMovement((0, 0, -movSpeed), False)

    if bge.logic.KX_INPUT_ACTIVE == keyboard.events[bge.events.UPARROWKEY]:
  #πανω
        hook_Mesh.applyMovement((0, 0, movSpeed), False)

    if bge.logic.KX_INPUT_ACTIVE ==
keyboard.events[bge.events.LEFTARROWKEY]:   #αριστερα
        hook_Mesh.applyMovement((-movSpeed, 0, 0), False)
```

Εικόνα 119 – ρυθμίσεις στον Logic Editor για το αγκίστρι και το Script που χρησιμοποιεί.

d. Fish

Στην Εικόνα 120, βλέπουμε τις εντολές για όλες τις ενέργειες που θέλουμε να κάνει.

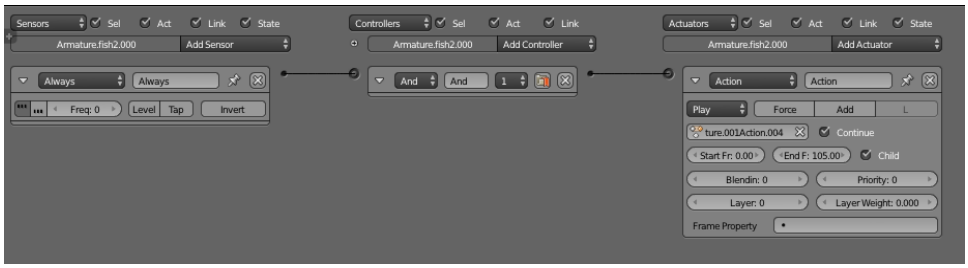
- Με τον Always Sensor και τον Motion Actuator οριζούμε μία συνεχόμενη κίνηση, κατά 0,20 μονάδες στον τοπικό άξονα y του αντικειμένου, και μία συνεχόμενη περιστροφή κατά 1^{o} στον τοπικό άξονα z.
- Στον Collision Sensor, βάζουμε το Material glass δηλαδή τη γυάλα, λέγοντας έτσι ότι θα υπάρχει κάποια αντίδραση όταν ακουμπήσει το αντικείμενο αυτό. Η αντίδραση θα είναι η περιστροφή 45^{o} στον local άξονα y, που το ορίζουμε με έναν Motion Actuator, στον οποίο επιλέγουμε τον τύπο Character Motion.
- Με τον Random Sensor και το πεδίο Seed ρυθμισμένο με μία τιμή εκτός του 0, δημιουργούμε τυχαίους παλμούς με τους οποίους κάνουμε το αντικείμενο Bubbles να βγαίνει απο το αντικείμενο Fish. Ο Actuator που χρησιμοποιούμε είναι ο Edit Object τον οποίο κάνουμε Add Object. Στο πεδίο Object βάζουμε το Bubbles, ρυθμίζουμε το Time για τον χρόνο το οποίο θα εμφανίζεται και ορίζουμε και μία ταχύτητα 50 στον Global άξονα z, ώστε να δώσουμε μία κίνηση προς τα πάνω.
- Ο sensor Near είναι για το Hook-Mesh και το Kamaki. Στο πεδίο Property έχουμε βάλει το Hook. Όποτε ακουμπάει το Fish, έχουμε δύο ενέργειες, άρα δύο Actuators. Ο ένας Actuator είναι ο Message στον οποίο βάζουμε σαν subject το μήνυμα points, το οποίο θα καλέσουμε για το μέτρημα των πόντων και την παραγωγή ενός ήχου, όποτε ένα αντικείμενο όπως το Fish το ακουμπάει το Hook-Mesh. Ο δεύτερος Actuator είναι ο Edit Object, τον οποίο έχουμε ρυθμίσει σε End Object, όταν δηλαδή θα το ακουμπάει, το Fish θα εξαφανίζεται.



Εικόνα 120 – ρυθμίσεις Logic Editor για το Fish

e. Armature

Η εικόνα 121, δείχνει τις εντολές που χρησιμοποιούνται για το Armature στον Logic Editor. Σαν Actuator, έχουμε βάλει τον Action και έχουμε επιλέξει το Play, και το όνομα του Armature του αντικείμένου. Πολύ σημαντικό είναι να βάλουμε στο Start και Stop, τα frames τα οποία έχουμε ορίσει κατά το Rigging του Armature.



Εικόνα 121 - ρυθμίσεις Logic Editor για το Fish Armature

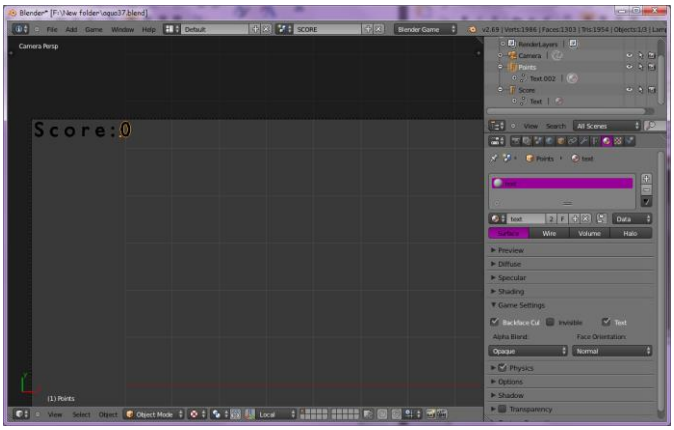
Την ίδια διαδικασία, κάνουμε για κάθε Armature, που έχουμε χρησιμοποιήσει, στο παιχνίδι.

f. Text – μετρητής πόντων

Τα Texts είναι αντικείμενα, όπως τα Béziers curves τα οποία περιέχουν κείμενο. Επιτρέπουν να δημιουργηθεί δισδιάστατο ή τρισδιάστατο κείμενο, με δυνατότητα μετακίνησης, περιστροφής και όλων των ιδιοτήτων που έχουν τα υπόλοιπα αντικείμενα. Σε Edit Mode μπορούμε να αλλάξουμε το περιεχόμενο του κειμένου. Με **alt C** σε Object Mode μπορούμε να μετατρέψουμε ένα Text σε Mesh.

Θα χρησιμοποιήσουμε το Text για το μέτρημα των πόντων του παιχνιδιού. Αρχικά θα δημιουργήσουμε μία νέα σκηνή την οποία θα ονομάσουμε SCORE, σε αυτήν θα προσθέσουμε μία νέα κάμερα η οποία θα κοιτάει κάτω. Προσθέτουμε ένα Text το οποίο σε Edit Mode θα αλλάξουμε σε "Score:", δίπλα θα βάλουμε ένα νέο Text που θα είναι το Points και το οποίο θα αλλάξουμε σε 0. Και τα δύο θα τα μετακινήσουμε πάνω αριστερά όπως βλέπουμε απο την κάμερα.

Στον Properties Editor στην καρτέλα Material, δημιουργούμε ένα Material και για τα δύο Texts και τσεκάρουμε το Text στο Game Settings. Εικόνα 122

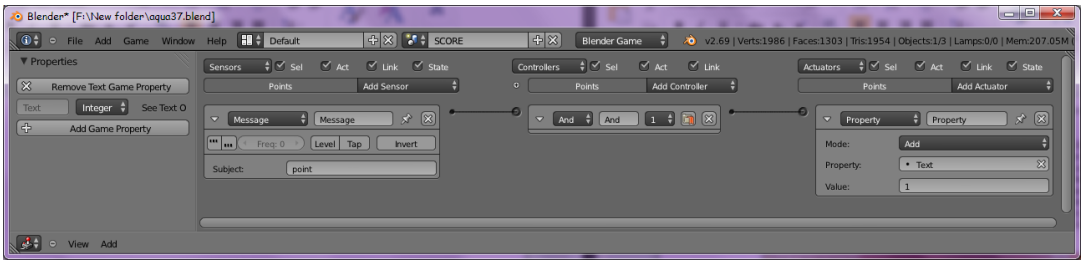


Εικόνα 122 – δημιουργία σκηνής για το μέτρημα των πόντων

Στον Logic Editor, θα κάνουμε το Text Points να αυξάνεται κατά ένα, κάθε φορά που το Hook-Mesh αγγίζει ένα Fish.

Όπως φαίνεται στην Εικόνα 123, θα χρησιμοποιήσουμε το Tab Properties για να προσθέσουμε ένα Text Game Property το οποίο έχει το όνομα Text και είναι ένας ακέραιος αριθμός.

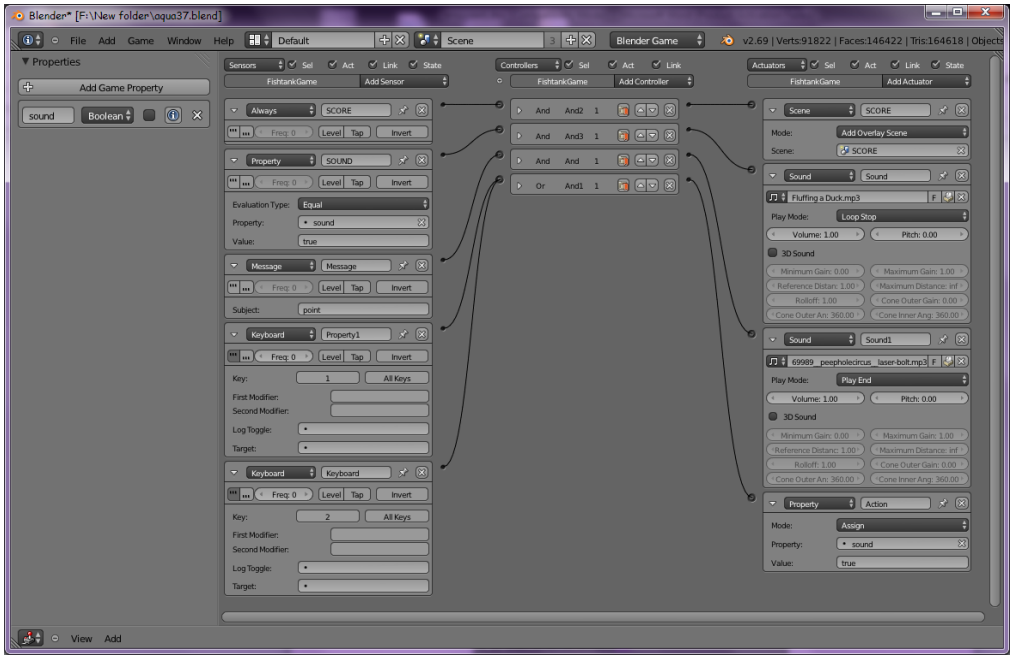
Θα χρησιμοποιήσουμε έναν Message Sensor στον οποίο θα βάλουμε σαν Subject το point. Έτσι θα ενεργοποιεί έναν Property Actuator στον οποίο αλλάζουμε το Mode σε Add, βάζουμε στο Property το Text και στο Value το 1, δηλαδή ορίζουμε να γίνεται πρόσθεση κατά 1, στο ήδη υπάρχον Text.



Εικόνα 123 – ρυμθίσεις Logic Editor για την αύξηση των points

g. Empty- FishtankGame

Χρησιμοποιούμε ένα Empty το οποίο ονομάζουμε FishtankGame. Στην Εικόνα 124 βλέπουμε τις ρυθμίσεις για τον Logic Editor.



Εικόνα 124 – ρυθμίσεις για FishtankGame

Αρχικά στο Properties Tab, έχουμε φτιάξει ένα Property με το όνομα sound, το οποίο είναι Boolean και έχει αρχική τιμή false.

- Στον Always Sensor και τον Scene Actuator, έχουμε ρυθμίσει το Mode σε Add Overlay Scene, και στο Scene έχουμε βάλει την σκηνή SCORE. Έτσι πάντα όταν τρέχουμε το παιχνίδι, θα φαίνονται πάνω αριστερά στην οθόνη τα δύο Texts της σκηνής SCORE.
- Με τους δύο Keyboard Sensors, όποτε πατάμε το 1 ή το 2, δηλαδή όποτε μεταφερόμαστε αντίστοιχα στις Camera01 και Camera02, αναπαράγεται ένας ήχος. Αυτό το ορίζουμε απο τον Property Actuator στον οποίο βάζουμε στο πεδίο Property το sound που φτιάξαμε παραπάνω και στο value γράφουμε true.
- Βάζουμε έναν Property Sensor. Στο πεδίο Property, βάζουμε το sound, στο value γράφουμε true. Όποτε το sound είναι αληθές τότε αναπαράγεται ένας ήχος, τον οποίο βάζουμε με τον Sound Actuator. Ανοίγουμε ένα αρχείο ήχου, και το Play Mode το κάνουμε Loop Stop. Έτσι συνδέουμε τον ήχο με τις δύο παραπάνω κάμερες.
- Στον Message Sensor βάζουμε στο subject την τιμή point, που χρησιμοποιούμε όποτε το Hook-Mesh αγγίζει ένα Fish. Όποτε γίνεται αυτό, με έναν Sound Actuator, βάζουμε έναν νέο ήχο να ακούγεται κάθε φορά. Στο Play Mode επιλέγουμε το Play End.

i. Python script για την κίνηση της Camera01 [65]

```
from bge import logic as G [1]
from bge import render as R
from bge import events
from mathutils import *

scene = G.getCurrentScene() [2]

obj = scene.objects["Empty01"] [3]

speed = 0.01 [4]
sensitivity = 0.0003
smooth = 0.7

cont = G.getCurrentController() [5]
owner = cont.owner
Mouse = cont.sensors["Mouse"] [6]

w = R.getWindowWidth()//2 [7]
h = R.getWindowHeight()//2
screen_center = (w, h)

keyboard = bge.logic.keyboard [8]
JUST_ACTIVATED = bge.logic.KX_INPUT_JUST_ACTIVATED

if "oldX" not in owner: [9]

        R.setMousePosition(w + 1, h + 1)

        owner["oldX"] = 0.0
        owner["oldY"] = 0.0

else: [10]

        scrc = Vector(screen_center)
        mpos = Vector(Mouse.position)

        x = scrc.x-mpos.x
        y = scrc.y-mpos.y

        owner['oldX'] = (owner['oldX']*smooth + x*(1.0-smooth)) [11]
        owner['oldY'] = (owner['oldY']*smooth + y*(1.0-smooth))

        x = owner['oldX']* sensitivity [12]
        y = owner['oldY']* sensitivity

        owner.applyRotation([0, 0, x], True) [13]
        owner.applyRotation([0, -y, 0], True)
```

```
        R.setMousePosition(*screen_center) [14]

if keyboard.events[bge.events.RKEY] == JUST_ACTIVATED: [15]
        owner.worldOrientation = [0, 0, 0]
```

1. Εισαγωγή των βιβλιοθηκών με τις κλάσεις που θα χρησιμοποιήσουμε, ώστε το script να μπορεί να αλληλεπιδράσει μαζί τους.
   Το Logic αναφέρεται στην κλάση των λειτουργιών του Game Logic, το Render στην κλάση για την απόδοση του παιχνιδιού.
   Το Events για τη χρήση των sensors, και το Mathutils παρέχει πρόσβαση σε μαθηματικές μεθόδους.
   Τα G και R είναι σταθερές τις οποίες ορίσαμε ώστε να μπορούμε να αναφερόμαστε στις κλάσεις Logic και Render πιο εύκολα.
   Το bge σημαίνει ότι μπορούμε να έχουμε πρόσβαση στις λειτουργίες του Blender Game Engine.
2. Ορίζει την τρέχουσα σκηνή που θα χρησιμοποιηθεί και της δίνουμε το όνομα scene.
3. Ορίζουμε το αντικείμενο της σκηνής που θα χρησιμοποιήσουμε και το τοποθετούμε στην μεταβλητή obj. Άρα το obj αντιπροσωπεύει το Empty01.
4. Ορίζουμε τρείς μεταβλητές και τους δίνουμε τιμές.
   Η speed είναι για την ταχύτητα που θα έχει η κάμερα, η sensitivity την γρήγορη ανταπόκριση στις κινήσεις του ποντικιού και η smooth την ομαλότητα ώστε να μοιάζει ρεαλιστική η κίνηση.
5. Γενική λειτουργία η οποία παίρνει τον controller της Python τον συνδέει με το συγκεκριμένο script της Python και το αποτέλεσμα το αποθηκεύει στην μεταβλητή cont.
   Στη μεταβλητή owner ορίζουμε τον παραπάνω controller, είναι αυτή που εκπροσωπεί το αντικείμενο στο οποίο βάζουμε το script δηλαδή στο Empty01.
6. Δίνουμε τιμή στη μεταβλητή με το όνομα Mouse, το όνομα του sensor που είναι συνδεδεμένο στον controller.
7. Για το rendering ορίζουμε δύο μεταβλητές.
   Στην w δίνουμε την ακέραια τιμή σε pixel του πλάτους του παραθύρου που τρέχει το παιχνίδι διά του 2.
   Στην h δίνουμε την αντίστοιχη τιμή του ύψους του παραθύρου.
   Στην μεταβλητή screen_center δίνουμε την τιμή που αντιστοιχεί στο κέντρο του παραθύρου αυτού.
   Στην ουσία χωρίζουμε το παράθυρο στη μέση έτσι ώστε να ξέρουμε τους άξονες και πάνω σε αυτούς να ορίσουμε τις κινήσεις του mouse.
8. Θα χρησιμοποιήσουμε σαν sensor και το πληκτρολόγιο. Η μεταβλητή keyboard θα χειρίζεται την είσοδο της εντολής. Η JUST_ACTIVATED είναι μεταβλητή η οποία ενημερώνει πότε μία κατάσταση είναι ενεργοποιημένη.
9. Ορίζουμε δύο προσωρινές μεταβλητές, τις oldX και oldY.
   Η oldX εννοεί την οριζόντια θέση του ποντικιού και η oldY την κάθετη.
   Με την if ελέγχουμε αν η oldX δεν ανήκει στην owner, ελέγχουμε δηλαδή αν η οριζόντια θέση του ποντικιού, δεν τρέχει με τον controller στο script.
   Αν είναι σωστό, τότε η θέση του mouse κατά το τρέξιμο του παιχνιδιού, θα αυξηθεί κατά ένα pixel και στον οριζόντιο και στον κάθετο άξονα.
   Επίσης οι μεταβλητές oldX και oldY θα ανήκουν πλέον στην owner, άρα και η οριζόντια και η κάθετη θέση του ποντικιού θα τρέχουν με τον controller στο script.
10. Αν είναι λάθος, δηλαδή αν ανήκει η oldX στην owner, τότε δίνει στην μεταβλητή scrc την διανυσματική τιμή του screen_center.
    Επίσης στη μεταβλητή mpos,δίνεται η διανυσματική τιμή του Mouse.position.
    Μία νέα μεταβλητή x παίρνει την τιμή της διανυσματικής τιμής του screen_center μείον της διανυσματικής τιμής του Mouse.position, το ίδιο γίνεται αντίστοιχα και σε μιά νέα y.
11. Οι δύο εντολές έχουν να κάνουν με την ομαλότητα της κίνησης του mouse στους δύο άξονες.

12. Εντολές για την ευαισθησία του mouse όταν κινείται.
13. Η μεταβλητή owner περιστρέφεται σύμφωνα με την τιμή της x στον άξονα z του σύστηματος αξόνων του χώρου. Το False μας δείχνει ότι οι άξονες είναι Global.
    Η yaw περιστρέφεται σύμφωνα με την τιμή της y στον x άξονα, ενώ το True μας λέει οτι έχουμε Local προσανατολισμό αξόνων.
14. Η θέση που θα έχει το mouse κατά το τρέξιμο του παιχνιδιού θα είναι στο κέντρο του παραθύρου.
15. Αν ενεργοποιηθεί η κατάσταση στην οποία δεχόμαστε είσοδο απο το πληκτρολόγιο το πλήκτρο R, δηλαδή αν πατήσουμε το R, τότε το owner, δηλαδή το Empty01 θα περιστραφεί στους άξονες στη θέση 0,0,0.

ii. Python script για την κίνηση του Hook [66]

```
import bge

def main(): 1
    1.
    cont = bge.logic.getCurrentController() 2

    hook_Mesh = cont.owner 3

    keyboard = bge.logic.keyboard 4

    movSpeed = 0.1 5


    if bge.logic.KX_INPUT_ACTIVE == keyboard.events[bge.events.DOWNARROWKEY]: 6
        hook_Mesh.applyMovement((0, 0, -movSpeed), False)

    if bge.logic.KX_INPUT_ACTIVE == keyboard.events[bge.events.UPARROWKEY]:
        hook_Mesh.applyMovement((0, 0, movSpeed), False)

    if bge.logic.KX_INPUT_ACTIVE == keyboard.events[bge.events.LEFTARROWKEY]:
        hook_Mesh.applyMovement((-movSpeed, 0, 0), False)

    if bge.logic.KX_INPUT_ACTIVE == keyboard.events[bge.events.RIGHTARROWKEY]:
        hook_Mesh.applyMovement((movSpeed, 0, 0), False)

    if bge.logic.KX_INPUT_ACTIVE == keyboard.events[bge.events.AKEY]:
        hook_Mesh.applyMovement((0, movSpeed, 0), False)

    if bge.logic.KX_INPUT_ACTIVE == keyboard.events[bge.events.ZKEY]:
        hook_Mesh.applyMovement((0, -movSpeed, 0), False)

main()
```

1. Χρησιμοποιούμε μία main() για να γράψουμε το script
2. Γενική λειτουργία η οποία παίρνει τον controller της Python τον συνδέει με το συγκεκριμένο script της Python και το αποτέλεσμα το αποθηκεύει στην μεταβλητή cont
3. Το αντικείμενο όπου θα εφαρμοστεί στο script είναι το hook_mesh
4. Χρήση του πληκτρολογίου σαν Sensor
5. Δίνουμε τιμή σε μία μεταβλητή την movSpeed, την οποία θα χρησιμοποιήσουμε σαν ταχύτητα
6. Αν ενεργοποιηθεί είσοδος από το πληκτρολόγιο με το πλήκτρο down arrow, δηλαδή αν πατηθεί το συγκεκριμένο πλήκτρο, τότε θα εφαρμοστεί κίνηση στο hook_mesh στον z άξονα όσο είναι και η τιμή της movSpeed. Το false δηλώνει ότι ο προσανατολισμός του αντικειμένου γίνεται καθολικά και όχι τοπικά.
   Σύμφωνα με το παραπάνω, η πρώτη εντολή μας λέει ότι με το down arrow το αντικείμενο θα μετακινηθεί προς τα κάτω, η δεύτερη ότι με το up arrow το αντικείμενο θα μετακινηθεί προς τα πάνω, με το left arrow προς τα αριστερά, με το right arrow προς τα δεξιά, με το A προς τα μέσα και με το Z προς τα έξω.