ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΚΑΙ ΠΛΟΗΓΗΣΗ ΣΕ ΑΛΛΗΛΕΠΙΔΡΑΣΤΙΚΟΥΣ ΕΙΚΟΝΙΚΟΥΣ ΚΟΣΜΟΥΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Αικατερίνη Μ. Αντωνοπούλου

Επιβλέπων : Στέφανος Κόλλιας
Καθηγητής Ε.Μ.Π

Αθήνα, Ιούλιος 2006

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΚΑΙ ΠΛΟΗΓΗΣΗ ΣΕ ΑΛΛΗΛΕΠΙΔΡΑΣΤΙΚΟΥΣ ΕΙΚΟΝΙΚΟΥΣ ΚΟΣΜΟΥΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Αικατερίνη Μ. Αντωνοπούλου

Επιβλέπων : Στέφανος Κόλλιας
Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την …. Ιουλίου 2006

……	……..……	……
Στέφανος Κόλλιας	Ανδρέας Σταφυλοπάτης	Παναγιώτης Τσανάκας
Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π

Αθήνα, Ιούλιος 2006

Αικατερίνη Μ. Αντωνοπούλου
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π

ΠΕΡΙΛΗΨΗ

Η εφαρμογή που αναπτύχθηκε στα πλαίσια της παρούσας εργασίας και περιγράφεται αναλυτικά στα επόμενα κεφάλαια δίνει την δυνατότητα στον χρήστη να πλοηγηθεί σε έναν interactive εικονικό κόσμο που αναπαριστά ένα μουσείο μοντέρνας τέχνης και τον περιβάλλοντα χώρο. Για την ολοκλήρωση της εφαρμογής απαιτήθηκε ο σχεδιασμός των αντικειμένων και των κτιριακών δομών που χρησιμοποιούνται από τον εικονικό κόσμο, η δημιουργία του εικονικού κόσμου, η μοντελοποίηση των χαρακτήρων και ο σχεδιασμός των ακολουθιών κίνησης τους, η κατασκευή των interfaces που επιτρέπουν την επικοινωνία του χρήστη με την εφαρμογή και ο προγραμματισμός της συμπεριφοράς των χαρακτήρων και των αντικειμένων του εικονικού κόσμου καθώς και των δυνατοτήτων της εφαρμογής. Οι παραπάνω διεργασίες περιγράφονται αναλυτικά στα κεφάλαια που ακολουθούν. Τα θέματα που επεξεργάζεται το κάθε κεφάλαιο αναφέρονται συνοπτικά στη συνέχεια.

Το πρώτο κεφάλαιο αποτελεί μία σύντομη αναφορά στην χρήση της εικόνας στις σύγχρονες εφαρμογές, στα κύρια χαρακτηριστικά των αλληλεπιδραστικών εφαρμογών πολυμέσων και στους τομείς της σύγχρονης ζωής οπού χρησιμοποιούνται παρεμφερείς εφαρμογές σε μεγάλη κλίμακα.

Το δεύτερο κεφάλαιο αναφέρεται στα προγράμματα που χρησιμοποιήθηκαν για τον σχεδιασμό της εφαρμογής. Περιγράφονται οι κύριες δυνατότητες που παρέχουν τα προγράμμάτα αυτά και η συμβολή τους στα διαφορετικά στάδια του σχεδιασμού της εφαρμογής.

Στο τρίτο κεφάλαιο περιγράφεται η αρχιτεκτονική πελάτη\εξηπηρετητή (client\server), που χρησιμοποιήθηκε για την οργάνωση της εφαρμογής. Επίσης αναφέρονται οι λόγοι που οδήγησαν στην επιλογή του συγκεκριμένου μοντέλου οργάνωσης.

Το τέταρτο κεφάλαιο αποτελεί μία περιγραφή του περιβάλλοντος του εικονικού κόσμου και των αντικειμένων που το απαρτίζουν. Συγκεκριμένα γίνεται αναφορά στην κατασκευή του περιβάλλοντος των εξωτερικών χώρων, στην μοντελοποίηση των κτιρίων και των αντικειμένων και την εισαγωγή τους στον εικονικό κόσμο και στην κατασκευή αντικειμένων ελέγχου και αντικειμένων οργάνωσης της εφαρμογής

Το πέμπτο κεφάλαιο αναφέρεται στην μοντελοποίηση των χαρακτήρων και στον σχεδιασμό των ακολουθιών της κίνησης τους(animation). Επίσης περιγράφεται ο τρόπος εισαγωγής τους στον εικονικό κόσμο και ο προγραμματισμός της συμπεριφοράς τους. Αναλύονται οι τρεις κατηγορίες χαρακτήρων της εφαρμογής: οι χαρακτήρες των οποίων η κίνηση καθορίζεται από τον χρήστη, οι χαρακτήρες των οποίων η κίνηση καθορίζεται με χρήση τεχνικών τεχνητής νοημοσύνης (κίνηση πάνω σε προκαθορισμένα μονοπάτια) και οι ακίνητοι χαρακτήρες.

Στο έκτο κεφάλαιο περιγράφονται οι τρόποι με τους οποίους επιτυγχάνεται η αλληλεπίδραση του χρήστη με το περιβάλλον της εφαρμογής. Παρουσιάζονται οι callback συναρτήσεις και μέθοδοι, τα αντικείμενα της κλάσης Trigger και η διαδικασία του binding, που καθιστούν την εφαρμογή αλληλεπιδραστική(interactive).

Το έβδομο κεφάλαιο αναφέρεται στα Graphical User Interfaces (GUIs) που παρέχουν στον χρήστη έναν φιλικό τρόπο επικοινωνίας με την εφαρμογή. Περιγράφονται τα βασικότερα GUIs που χρησιμοποιεί η εφαρμογή στα διάφορα στάδια της εκτέλεσής της.

Το παράρτημα που παρατίθεται μετά το έβδομο κεφάλαιο περιλαμβάνει τον κώδικά της εφαρμογής.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ

Εικονικός κόσμος, game programming, Torque game engine, script language, μοντελοποίηση χαρακτήρων, σχεδιασμός ακολουθιών κίνησης (animation), σχεδιασμός αντικειμένων και κτιριακών δομών, αρχιτεκτονική πελάτη-εξυπηρετητή, αλληλεπιδραστικότητα (interactivity),εφαρμογές πολυμέσων(multimedia applications), εικόνα, video, γραφικές διαμεσολαβήσεις χρήστη (Graphical User Interfaces).

ABSTRACT

The application presented in this project offers the user the opportunity to visit an interactive virtual world that represents a museum of modern art and the environment space around the museum. The process of the implementation of the application required the design of the objects and the building structures used by the virtual world, the design of the environment of the virtual world, the modeling of the characters and the design of their animation, the construction of interfaces that allow the communication enter the user and the application as well as the programming of, application and the determination of the characters and the objects' behavior in the virtual world. The process described above is explained analytically in the chapters that follow. Here takes place a brief report of the contents of each chapter

The first chapter constitutes a brief reference of the role of the image in the modern applications, the main characteristics of interactive multimedia applications and the role of these applications in various sectors of modern life.

The second chapter refers to the programs that were used in the implementation of the application. We describe the main possibilities that provide each program and its contribution in the different stages of the construction of the application.

In the third chapter we describe the client\server architecture used by the application. Also we explain the reasons that led to the choice of the particular model.

The fourth chapter constitutes a description of the environment of the virtual world. We describe the design of the exterior space, the modeling of buildings and objects and the way they were imported in the visual world. Also we explain the use of objects that contribute in the control and organization of the application.

The fifth chapter refers to the modeling of characters and the design of their animation. Also we describe the way they were imported in the virtual world and the programming of their behavior. We mention the three categories of characters used in the application: the characters whose movement is determined by the user, the characters whose movement is determined by the programmer with use of techniques of artificial intelligence (movement on predetermined paths) and the motionless characters.

In the sixth chapter we describe the implementation of the interaction of the user with the environment of the application. We present the callback functions and methods, the objects of the class Trigger and the process of binding, that make the application interactive.

The seventh chapter refers to the Graphical User Interfaces (GUIs), that constitute a friendly way of communication enter the user and the application. We present the different GUIs that are used by the application in the various stages of its implementation.

The appendix at the end consists of the code of the application.

KEYWORDS

Virtual world, game programming, Torque game engine, script language, characters modeling, animation, design of object and building structures, client-server architecture, interactivity, multimedia applications, image, video, Graphical User Interfaces.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΙΝΑΚΑΣ ΣΧΗΜΑΤΩΝ, ΕΙΚΟΝΩΝ ΚΑΙ ΠΙΝΑΚΩΝ

1. ΕΙΣΑΓΩΓΗ.

Η εικόνα ως μέσο αναπαράστασης της πραγματικότητας, επέκτασής της, ακόμα και υποκατάστασής της; έχει κατοχυρώσει την κυριαρχία της σε πολλούς τομείς της σύγχρονης ζωής. Ένα μεγάλο πλήθος εφαρμογών χρησιμοποιεί την εικόνα ως μέσο μεταφοράς πληροφορίας.

Η πρόοδος της επιστήμης των υπολογιστών έχει συμβάλει στην ανάπτυξη εφαρμογών που χρησιμοποιούν κινούμενη ή στατική εικόνα. Μέσω της ψηφιακής αναπαράστασης της πληροφορίας και της ολοκλήρωσης διαφορετικών τύπων συσκευών στους Η\Υ έχει επιτευχθεί η ενσωμάτωση διαφορετικών τύπων πληροφορίας στην ίδια εφαρμογή. Έτσι συνεχώς αναπτύσσονται νέες πολυμεσικές εφαρμογές που συνδυάζουν πολλαπλούς τύπους πληροφορίας όπως εικόνα, ήχο κείμενο και γραφικά.

Οι εφαρμογές αυτές χρησιμοποιούνται σε μεγάλο βαθμό στους χώρους της εργασίας, της διασκέδασης, της εκπαίδευσης, της ενημέρωσης, της διαφήμισης, της επικοινωνίας και της ιατρικής. Ιδιαίτερα στον τομέα της εκπαίδευσης η χρήση πολυμεσικών εφαρμογών κάνει την διαδικασία της μάθησης πιο ευχάριστη και ενδιαφέρουσα για τον εκπαιδευόμενο, ειδικά όταν απευθύνεται σε παιδιά μικρής ηλικίας. Επιπλέον ο συνδυασμός εικόνας και ήχου διευκολύνει την αφομοίωση της παρεχόμενης πληροφορίας και μετατροπή της σε γνώση.

Το game programming αποτελεί ένα σχετικά νέο κλάδο που έχει αναπτυχθεί ταχύτατα τα τελευταία χρόνια και οι εφαρμογές του έχουν κατακλύσει τον χώρο της ψυχαγωγίας. Πέραν όμως από ένα μέσο σχεδιασμού video games, το game programming μπορεί να αποτελέσει και ένα πολύ ισχυρό εργαλείο για την σχεδίαση μιας πλειάδας άλλων πολυμεσικών εφαρμογών. Στο πεδίο της εκπαίδευσης έχει ενσωματωθεί πλήθος εφαρμογών που περιλαμβάνουν πλοήγηση σε interactive εικονικούς κόσμους, όπως η πλοήγηση στον χώρο του εικονικού μουσείου.

Για την σχεδίαση ενός εικονικού κόσμου απαιτείται μια σειρά διεργασιών που αφορούν στον σχεδιασμό του περιβάλλοντος, των κτιρίων και των αντικειμένων που εμφανίζονται στον εικονικό κόσμο. Επίσης απαιτείται η μοντελοποίηση των χαρακτήρων, ο σχεδιασμός της κίνησής τους και ο προγραμματισμός της συμπεριφοράς τους. Τέλος απαιτείται η κωδικοποίηση της αλληλεπίδρασης και των άλλων δυνατοτήτων της εφαρμογής.

Ιδιαίτερη έμφαση δίνεται στην δυνατότητα αλληλεπίδρασης μεταξύ χρήστη και εφαρμογής. Η αλληλεπίδραση επιτρέπει στον χρήστη να καθορίζει τον τρόπο παρουσίασης της πληροφορίας σύμφωνα με τις ανάγκες του. Μία interactive εφαρμογή μπορεί να παρέχει στον χρήστη την δυνατότητα να καθορίσει την σειρά, την ταχύτητα και την μορφή παρουσίασης της πληροφορίας. Οι τρεις αυτοί παράγοντες που ονομάζονται βαθμοί προσαρμοστικότητας στις επιθυμίες του χρήστη συμβάλλουν στην «συνεργασία» χρήστη-εφαρμογής.

Εξίσου σημαντικό στοιχείο μίας εφαρμογής είναι η φιλικότητα της προς τον χρήστη. Η κατασκευή των κατάλληλων interfaces κάνουν την εφαρμογή προσιτή σε περισσότερες ομάδες χρηστών και διευκολύνουν την επικοινωνία μεταξύ χρήστη και εφαρμογής. Έτσι μέσω απλών, γραφικών interfaces μπορεί να δοθεί η δυνατότητα προσπέλασης μίας εφαρμογής από ανθρώπους που δεν έχουν εξειδικευμένες γνώσεις στην χρήση των υπολογιστών. Τα interfaces αυτά με εύκολο και κατανοητό τρόπο καθοδηγούν τον χρήστη και τον ενημερώνουν για τις δυνατότητες που του παρέχει η εφαρμογή.

Επιπλέον, πολλές σύγχρονες εφαρμογές σχεδιάζονται με τρόπο που να επιτρέπει την ταυτόχρονη προσπέλαση της εφαρμογής από πολλούς χρήστες. Η αρχιτεκτονική πελάτη \εξυπηρετητή (client\server architecture) υποστηρίζει αυτή τη δυνατότητα. Σύμφωνα με το μοντέλο client\server ο εξυπηρετητής μπορεί να επικοινωνεί και να ικανοποιεί τα αιτήματα όλων των πελατών με τους οποίους έχει συνδεθεί. Η σύνδεση πραγματοποιείται ύστερα από αίτημα του πελάτη που γίνεται αποδεκτό από τον εξυπηρετητή. Με την εξέλιξη και την ευρεία χρήση των δικτύων η ταυτόχρονη προσπέλαση μίας εφαρμογής από πολλούς χρήστες είναι πολύ συνηθισμένη, ενώ συχνά παρέχονται και επιπλέον δυνατότητες επικοινωνίας μεταξύ τον χρηστών.

Οι δυνατότητες των πολυμεσικών εφαρμογών εξελίσσονται διαρκώς και επεκτείνονται. Η εκτεταμένη χρήση τους μπορεί να διευκολύνει πολλούς τομείς της σύγχρονης ζωής, αλλά και να καταστήσει ακόμα πιο δυσδιάκριτα τα όρια μεταξύ θεάματος και πραγματικότητας.

«Το θέαμα είναι το κακό όνειρο της σύγχρονης αλυσοδεμένης κοινωνίας, που τελικά δεν εκφράζει παρά την επιθυμία της να κοιμηθεί.»(Guy Debord, Η κοινωνία του θεάματος)

2. ΕΡΓΑΛΕΙΑ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ.

Το πρόγραμμα που είναι υπεύθυνο για την εκτέλεση της εφαρμογής, που επιτρέπει την πλοήγηση στον εικονικό κόσμο του μουσείου, είναι το Torque Game Engine. Το πρόγραμμα αυτό αναλαμβάνει την μεταγλώττιση και φόρτωση του κώδικα της εφαρμογής και όλων των απαραίτητων αρχείων.

Πέραν του Torque Game Engine χρησιμοποιήθηκε μία σειρά άλλων προγραμμάτων για τον σχεδιασμό των αντικειμένων που απαρτίζουν τον εικονικό κόσμο. Τα κυριότερα από αυτά είναι τα προγράμματα Quake Army Knife (QuArK), MilkShape, CharacterFX και Adobe Photoshop που περιγράφονται παρακάτω. Τα προγράμματα αυτά χρησιμοποιήθηκαν για τον σχεδιασμό των κτιρίων, τον σχεδιασμό των αντικειμένων, την μοντελοποίηση των χαρακτήρων, την κατασκευή της κίνησης των χαρακτήρων και την επεξεργασία των texture.

Επιπλέον χρησιμοποιήθηκαν βοηθητικά προγράμματα όπως το TorqueShowTool Pro που χρησιμοποιήθηκε για την δοκιμή της κίνησης των χαρακτήρων πριν την εισαγωγή τους στον εικονικό κόσμο και το UVMapper που χρησιμοποιήθηκε την κατασκευή του skin των χαρακτήρων.

2.1 Torque Game Engine

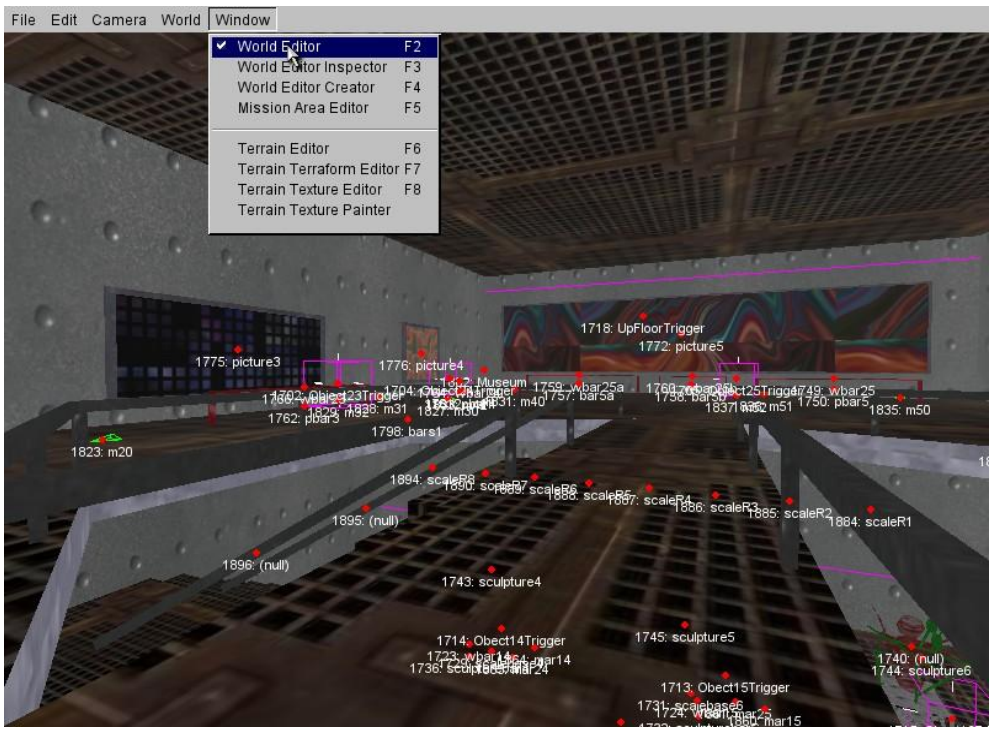
Όπως στην περίπτωση των περισσότερων Game Engines το Torque περιλαμβάνει τα παρακάτω τμήματα (modules): τμήμα τρισδιάστατων γραφικών, τμήμα ήχου, τμήμα τεχνητής νοημοσύνης, τμήμα ανίχνευσης σύγκρουσης, τμήμα εισόδου/εξόδου, τμήμα βάσης δεδομένων, τμήμα δικτύου και τμήμα γραφικών interfaces χρήστη. Το Torque υποστηρίζει την εγγραφή κώδικα σε script language. Η script language ακολουθεί το αντικειμενοστρεφές μοντέλο και υποστηρίζει τον ορισμό κλάσεων και datablocks. Κάθε αντικείμενο που κατασκευάζεται με χρήση της δεσμευμένης λέξης new ανήκει σε μία κλάση. Τα χαρακτηριστικά του αντικειμένου καθορίζονται από τις τιμές των πεδίων της κλάσης και η συμπεριφορά του από της μεθόδους που υλοποιεί. Ορισμένα αντικείμενα χρησιμοποιούν datablocks προκειμένου να αποκτήσουν επιπρόσθετες ιδιότητες ή να διαφοροποιηθούν από άλλα στιγμιότυπα της ίδιας κλάσης. Η script language υποστηρίζει επιπλέον την κληρονομικότητα και τον πολυμορφισμό. Το συντακτικό της γλώσσας μοιάζει με αυτό της C++ με κύρια διαφορά ότι στην script language δεν απαιτούνται δηλώσεις τύπων των μεταβλητών που χρησιμοποιούνται. Η script language κάνει διάκριση μεταξύ τοπικών και παγκόσμιων μεταβλητών και χρησιμοποιεί τις βασικότερες δομές ελέγχου ροής του προγράμματος και τους αριθμητικούς και λογικούς τελεστές, που χρησιμοποιεί και η C++. Επίσης υποστηρίζει δυναμικό φόρτωμα πακέτων του κώδικα όταν αυτό απαιτείται από την τρέχουσα εφαρμογή. Ο κώδικας του Torque περιλαμβάνει τον ορισμό περίπου 420 έτοιμων συναρτήσεων, 680 μεθόδων και 200 callbacks. Εκτός από τις προκαθορισμένες συναρτήσεις και μεθόδους του Torque δίνεται η δυνατότητα στον χρήστη να ορίσει και να χρησιμοποιήσει τις δικές του συναρτήσεις και μεθόδους.

Μέσω κώδικα γραμμένου σε script language μπορούν να ελεγχθούν όλες οι λειτουργίες των εφαρμογών που εκτελούνται στο Torque Game Engine, όπως η δημιουργία και καταστροφή αντικειμένων, ανίχνευση εισόδου και ανταπόκριση σε αυτή, ανίχνευση σύγκρουσης μεταξύ αντικειμένων, δημιουργία σύνδεσης σε

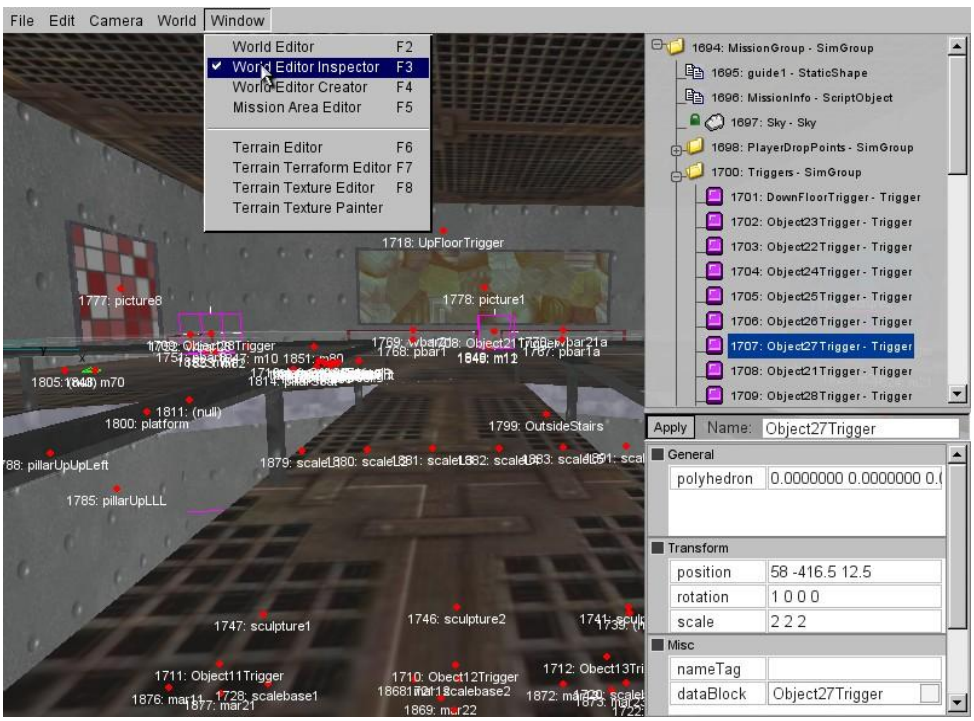
εφαρμογές που ακολουθούν την αρχιτεκτονική πελάτη/εξυπηρετητή, κατασκευή GUIs, καθορισμός θέσης, κλίσης και κλίμακας των αντικειμένων που εμφανίζονται στο περιβάλλον της εφαρμογής, καθορισμός animation και ταχύτητας κίνησης χαρακτήρων, χειρισμός ήχου, οργάνωση χρονοπρογραμμάτων και προσδιορισμός ακολουθιών διαδοχικών ενεργειών, συστήματα επικοινωνίας μεταξύ πολλαπλών χρηστών και προγραμματισμός ενεργειών που χρησιμοποιούν τις αρχές της τεχνητής νοημοσύνης.

Το Torque περιέχει δύο βασικούς χώρους εργασίας τον world editor και τον gui editor.

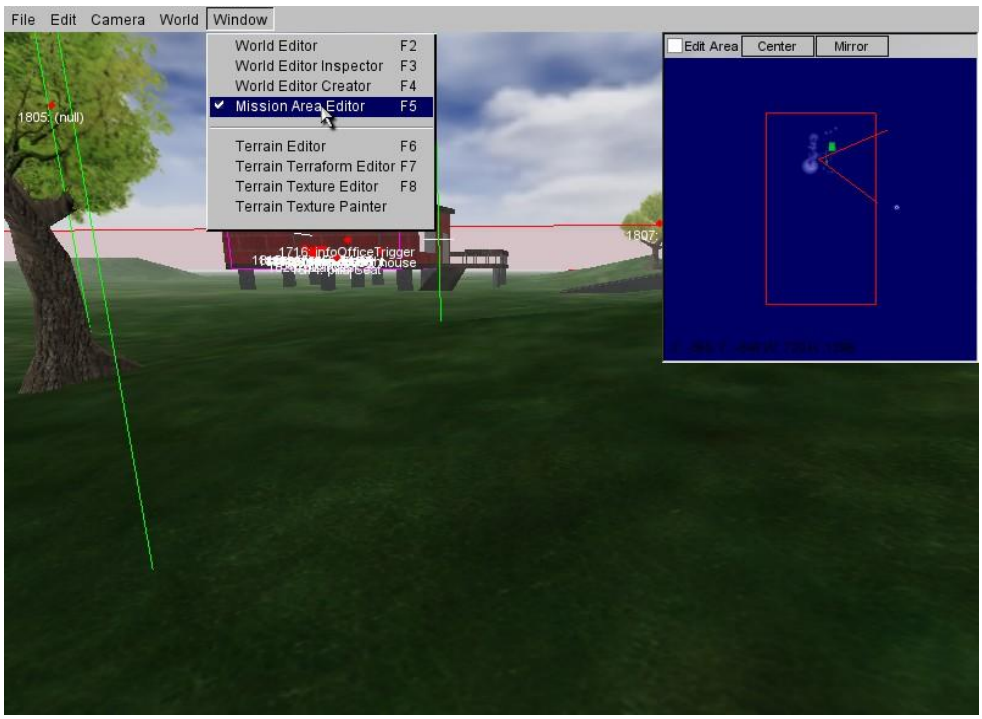
Ο world editor χρησιμοποιείται για τον σχεδιασμό του περιβάλλοντος της εφαρμογής και περιλαμβάνει τον terrain editor για την διαμόρφωση του terrain. Η επιλογή του world editor inspector εμφανίζει μια δενδρική δομή στο δεξί μέρος της οθόνης, όπου εμφανίζονται όλα τα αντικείμενα που αποτελούν το περιβάλλον του εικονικού κόσμου. Με επιλογή οποιουδήποτε αντικειμένου της δενδρικής δομής εμφανίζονται πληροφορίες που αφορούν στην θέση του, την κλίση του, το μέγεθός του και μία σειρά άλλων χαρακτηριστικών ιδιοτήτων που ποικίλουν ανάλογα με το είδος του αντικειμένου. Τα αντικείμενα του περιβάλλοντος μπορούν να ομαδοποιηθούν με χρήση αντικειμένων SimGroup που λειτουργούν σαν φάκελοι και συμβάλλουν στην καλύτερη οργάνωση των συστατικών στοιχείων του εικονικού κόσμου. Τα αντικείμενα που απαρτίζουν το περιβάλλον του εικονικού κόσμου κατασκευάζονται μέσω του world editor creator. Ο world editor creator χωρίζει τα αντικείμενα αυτά σε 4 κατηγορίες: αντικείμενα τύπου interiors, shapes, static shapes και mission objects. Μετά την κατασκευή ενός αντικειμένου δίνεται η δυνατότητα στον χρήστη να ορίσει την θέση που θα καταλαμβάνει στον εικονικό κόσμο, το όνομα του και τα υπόλοιπα χαρακτηριστικά του. Η επιλογή του mission area editor ανοίγει ένα παράθυρο που επιτρέπει μια γενική επισκόπηση του χώρου όπου μπορούν να τοποθετηθούν τα κτίρια και τα υπόλοιπα αντικείμενα του εικονικού κόσμου. Με την βοήθεια του mission area editor οριοθετείται ο χώρος που θα καταλαμβάνει το περιβάλλον του εικονικού κόσμου και καθορίζεται το μέγεθος του. Ο world editor παρέχει επιπλέον επιλογές που έχουν να κάνουν με την θέση, την λήψη και την ταχύτητα κίνησης της κάμερας και περιλαμβάνονται στο μενού camera. Μετά την τοποθέτηση ή την αλλαγή της θέσης οποιουδήποτε αντικειμένου είναι απαραίτητη η επιλογή του relight scene που εμφανίζει το texture του αντικειμένου και τις σκιές που δημιουργεί στο περιβάλλον του.



Εικόνα 2.1 World editor

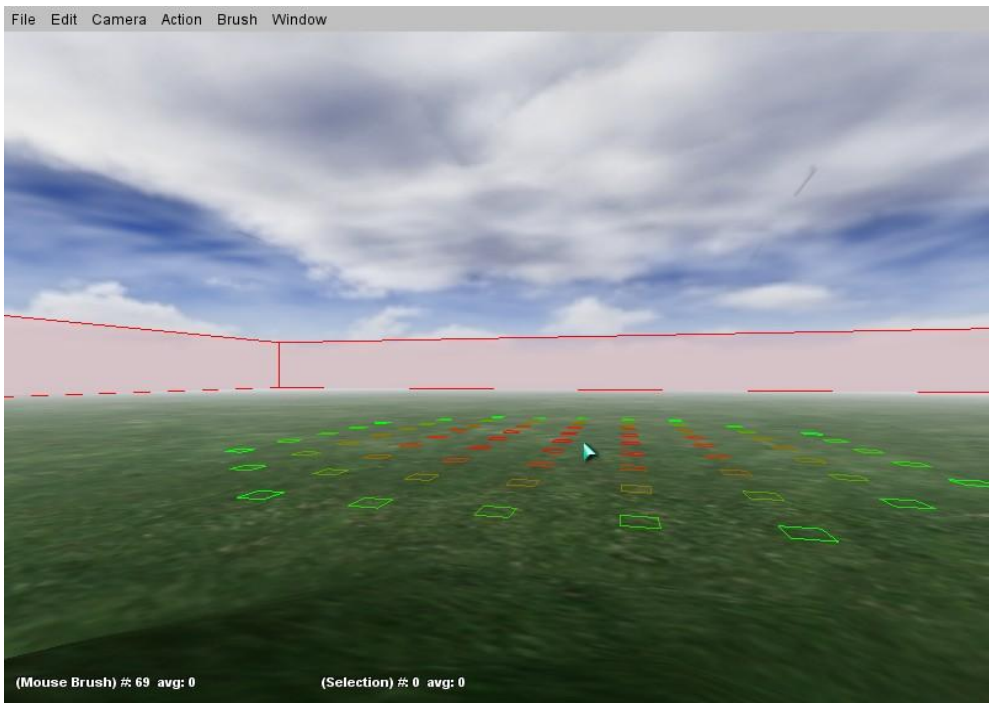


Εικόνα 2.2 World editor inspector



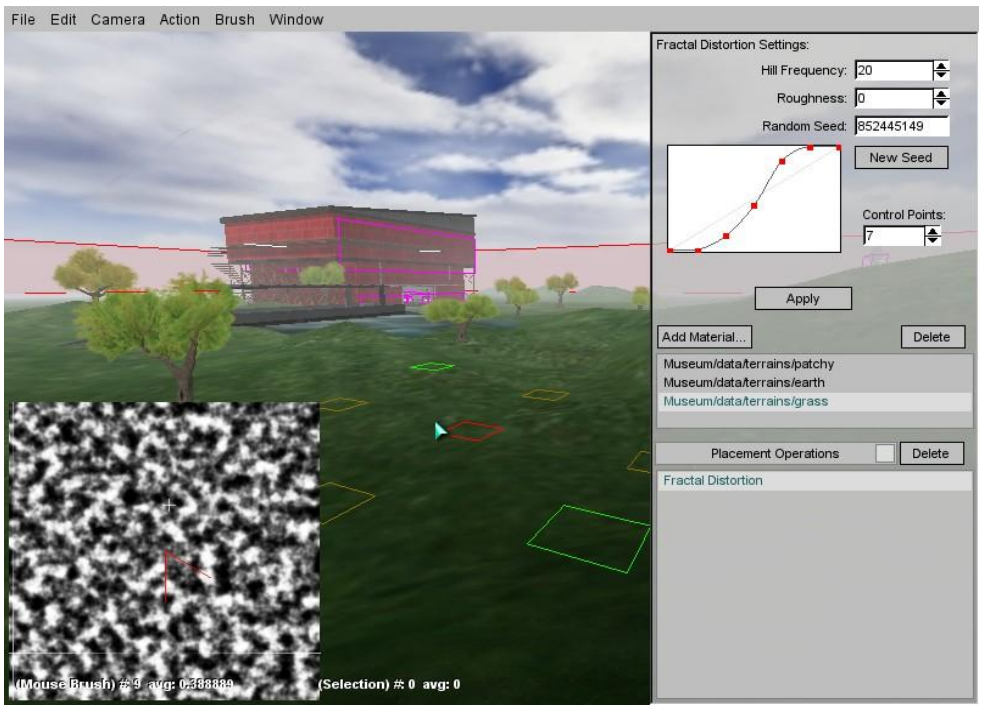
Εικόνα 2.3 Mission area editor

Ο terrain editor χρησιμοποιείται για την διαμόρφωση του terrain της εφαρμογής. Κατά την επιλογή του εμφανίζεται το εργαλείο brush που επιτρέπει την διαμόρφωση της τοπολογίας της εφαρμογής με δημιουργία υψωμάτων στο έδαφος. Το μέγεθος του εργαλείου brush, η ισχύς του και το σχήμα του ρυθμίζονται μέσω των επιλογών του μενού brush. Το εργαλείο terrain terraform editor χρησιμοποιείται για την αλγοριθμική επεξεργασία του terrain. Συγκεκριμένα ο terrain terraform editor παρέχει μια σειρά τελεστών και φίλτρων που αν εφαρμοστούν στο αρχικό terrain τροποποιούν την μορφή του. Για την εφαρμογή του texture του terrain υπάρχουν τα εργαλεία terrain texture editor και terrain texture painter. Συγκεκριμένα μέσω του terrain texture editor καθορίζεται ο τρόπος με τον οποίο θα εφαρμοστεί το texture στο terrain, προσδίδοντας στην εμφάνιση του περισσοτερες λεπτομέρειες. Ο terrain texture painter επιτρέπει την επιλογή μέχρι και 6 διαφορετικών texture τα οποία μπορούν να εφαρμοστούν σε διαφορετικά σημεία του terrain

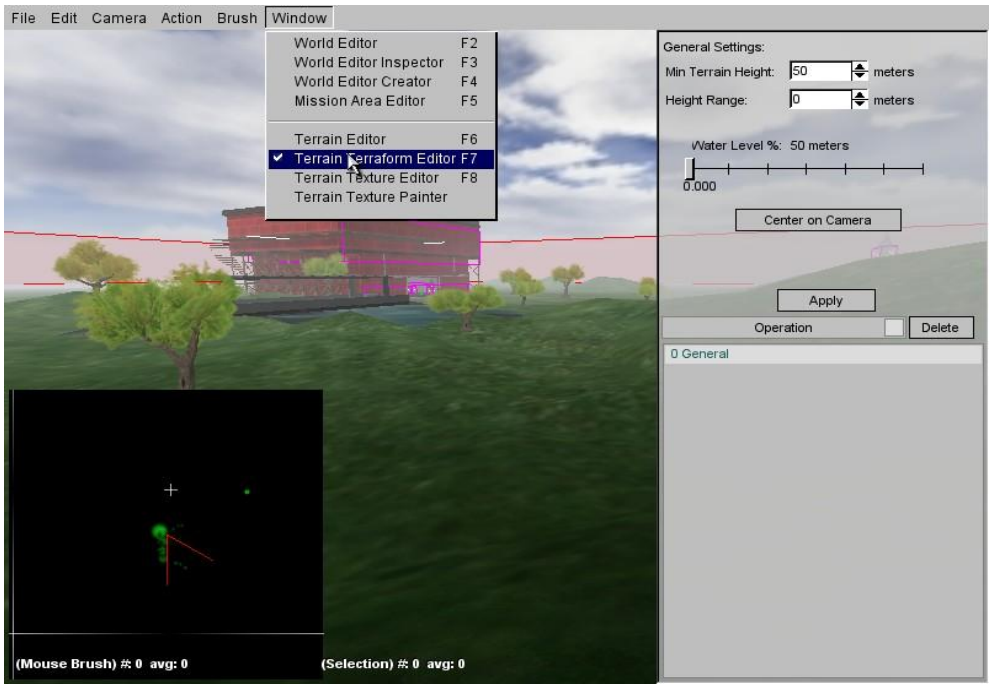


Εικόνα 2.4 Terrain editor

Εικόνα 2.5 Terrain texture painter



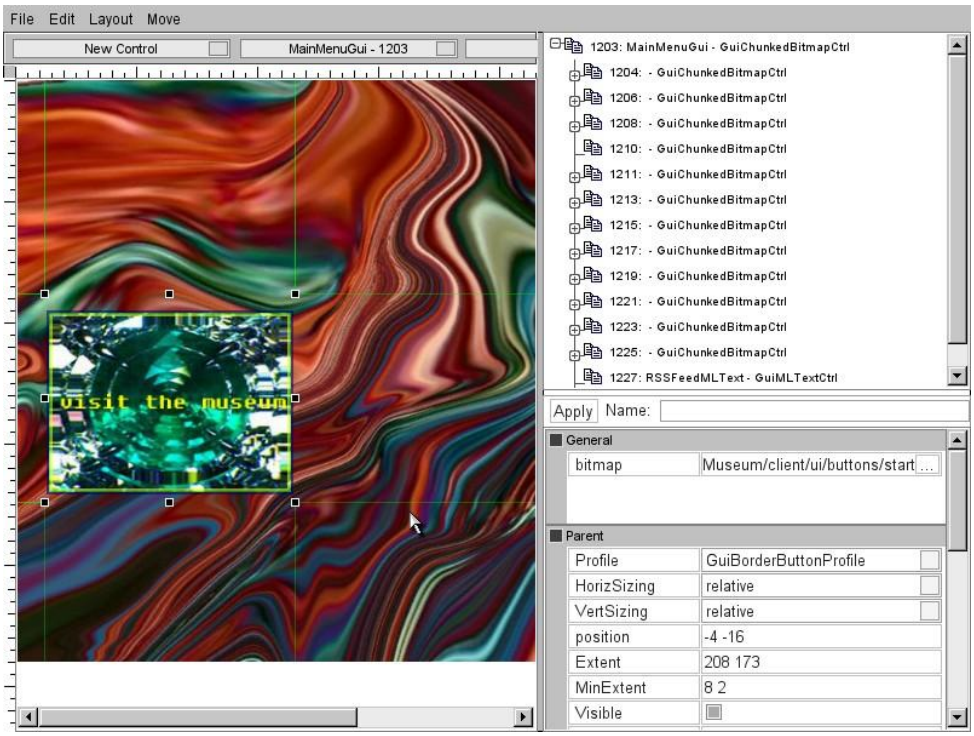
Εικόνα 2.6 Terrain texture editor



Εικόνα 2.7 Terrain terraform editor

Ο gui editor είναι ο χώρος εργασίας που επιτρέπει τον σχεδιασμό Graphical User Interfaces (GUIs) τα οποία χρησιμοποιούνται για την επικοινωνία του χρήστη με την εφαρμογή. Μέσω του gui editor δίνεται η δυνατότητα κατασκευής αντικειμένων που αποτελούν συστατικά στοιχεία των GUIs. Επίσης είναι δυνατός ο προσδιορισμός

της θέσης τους, του μεγέθους τους και των υπολοίπων χαρακτηριστικών τους όπως το αν θα είναι ορατά ή όχι καθώς και οι ενέργειες που θα εκτελούνται κατά την επιλογή τους, αν πρόκειται για αλληλεπιδραστικά αντικείμενα. Επίσης καθορίζεται το μέγεθος του παραθύρου της εφαρμογής και ο τρόπος με τον οποίο θα αναπροσαρμόζουν το μέγεθος και την θέση τους, τα συστατικά στοιχεία του GUI, όταν μεταβάλλεται το μέγεθος του παραθύρου που τα εμπεριέχει. Ο gui editor παρέχει μια δενδρική δομή στην οποία εμφανίζονται όλα τα αντικείμενα που απαρτίζουν το GUI. Με επιλογή κάποιου αντικειμένου της δενδρικής δομής εμφανίζονται οι πληροφορίες που αφορούν στο αντικείμενο αυτό.



Εικόνα 2.8 Gui editor

2.2 Quake Army Knife (QuArK)

Το πρόγραμμα QuArK χρησιμοποιήθηκε για τον σχεδιασμό των κτιρίων που περιέχονται στην εφαρμογή καθώς και τον σχεδιασμό άλλων αντικειμένων που χρησιμοποιήθηκαν στην εφαρμογή ως αντικείμενα τύπου interiors. Το πρόγραμμα αυτό περιλαμβάνει ένα πλήθος εργαλείων σχεδιασμού και texturing.

Το βασικό τμήμα του προγράμματος είναι ο map editor. Ο map editor

χρησιμοποιείται για τον σχεδιασμό των κτιρίων, των αντικειμένων και άλλων δομών. Περιλαμβάνει εργαλεία για τον σχεδιασμό απλών πολυέδρων, σύνθετων σχημάτων, καμπυλών και τόξων. Για τον χειρισμό και την μορφοποίηση των παραπάνω σχημάτων χρησιμοποιούνται μια σειρά εργαλείων που μεταξύ άλλων εκτελούν διεργασίες, όπως μετακίνηση αντικειμένων, περιστροφή γύρω από τους άξονες, προσαρμογή του μεγέθους των αντικειμένων, διαίρεση αντικειμένων ή επιφανειών. Για την ευκολότερη εκτέλεση των παραπάνω διεργασιών παρέχονται στον χρήστη επιλογές όπως η προσωρινή απόκρυψη ορισμένων οντοτήτων ή επιφανειών, η μετακίνηση αντικειμένων σε συγκεκριμένα προκαθορισμένα σημεία του χάρτη και η επιλογή αντικειμένων ή επιφανειών που ικανοποιούν συγκεκριμένες ιδιότητες. Το QuArK παρέχει μια συμπληρωματική σειρά εργαλείων για την περαιτέρω διευκόλυνση του σχεδιασμού. Τα βασικότερα από αυτά τα εργαλεία είναι οι dublicators, που διευκολύνουν την σχεδίαση επαναλαμβανόμενων δομών, καθώς και οι diggers για την ευκολότερη σχεδίαση εσοχών ή οπών στα αντικείμενα. Επιπλέον είναι δυνατή η εισαγωγή δομών όπως πηγές φωτός, επιφάνειες αντικατοπτρισμού, triggers, κόμβοι μονοπατιών και ένα πλήθος άλλων οντοτήτων. Ειδικά στην περίπτωση των πηγών φωτός παρέχεται μία μεγάλη ποικιλία επιλογής του είδους της πηγής και ρύθμιση των χαρακτηριστικών της, όπως το χρώμα του φωτός, η ένταση του και ο τρόπος εκπομπής του.

Η ταυτόχρονη επισκόπηση διαφορετικών όψεων του αντικειμένου είναι δυνατή με την εμφάνιση πολλαπλών παραθύρων. Το QuArK υποστηρίζει την εμφάνιση από δύο έως και τεσσάρων παραθύρων που παρουσιάζουν την πρόσοψη, την κάτοψη, την πλάγια όψη καθώς και την τρισδιάστατη μορφή της σχεδιαζόμενης δομής. Επίσης παρέχεται η δυνατότητα προεπισκόπησης της τελικής μορφής του σχεδιαζόμενου αντικειμένου μέσω της επιλογής full 3D layout που παρουσιάζει σε ξεχωριστό παράθυρο την τρισδιάστατη μορφή του αντικειμένου με τα texture που έχουν αντιστοιχηθεί στις επιφάνειές του.

Για την καλύτερη οργάνωση και την ευκολότερη επιλογή των συστατικών στοιχείων του χάρτη υπάρχει μία δενδρική δομή που περιέχει όλες τις σχεδιασμένες οντότητες και επιτρέπει την ομαδοποίηση τους . Μεσω της δενδρικής δομής είναι δυνατή η επιλογή κάθε αντικειμένου και η εμφάνιση των χαρακτηριστικών του επιλεγμένου αντικειμένου σε ξεχωριστό παράθυρο.

Η αντιστοίχηση texture στις επιφάνειες πραγματοποιείται με επιλογή της επιφάνειας και άνοιγμα του texture browser. Με την βοήθεια του texture browser είναι δυνατή η εισαγωγή εικόνων που αποτελούν τα texture και η αντιστοίχησή τους στις κατάλληλες επιφάνειες.

Οι χάρτες που σχεδιάζονται στο QuArK αποθηκεύονται ως αρχεία τύπου map ή qrk. Για την εισαγωγή των σχεδιασμένων αντικειμένων στο Torque είναι απαραίτητη η μετατροπή των αρχείων αυτών σε αρχεία τύπου dif. Η εξαγωγή σε αρχεία τύπου dif πραγματοποιείται με χρήση του κατάλληλου exporter που παρέχει το QuArK.

2.3 MilkShape

Το πρόγραμμα Milkshape χρησιμοποιήθηκε για τον σχεδιασμό αντικειμένων που περιλαμβάνονται στον εικονικό κόσμο της εφαρμογής καθώς και για την μοντελοποίηση των χαρακτήρων.

Τα αντικείμενα που σχεδιάζονται στο Milkshape αποτελούνται από ένα σύνολο σημείων-κορυφών και ένα σύνολο τριγωνικών επιφανειών. Ειδικά στην περίπτωση

των χαρακτήρων ή κινούμενων αντικειμένων υπάρχει και το σύνολο των κόμβων του σκελετού του αντικειμένου. Οι κορυφές και οι επιφάνειες που σχηματίζουν ένα αντικείμενο μπορούν να ομαδοποιηθούν για την διευκόλυνση του σχεδιαστή.

Ο σχεδιασμός των αντικειμένων πραγματοποιείται με χρήση των εργαλείων που παρέχει το πρόγραμμα και αφορούν στην κατασκευή και επιλογή κορυφών, επιφανειών, κόμβων ή ομάδων, την κίνηση τους, την περιστροφή τους, τον προσδιορισμό του μεγέθους τους, την διαίρεση ή την συνένωσή τους. Επίσης παρέχεται η δυνατότητα άμεσης κατασκευής απλών σχημάτων όπως ορθογωνίων παραλληλεπιπέδων, σφαιρών, κυλίνδρων ή απλών επιφανειών. Μερικές βασικές λειτουργίες που περιλαμβάνει το πρόγραμμα για την διαχείριση των κορυφών είναι η συνένωση δύο ή περισσότερων κορυφών, η δημιουργία κορυφής πάνω σε μία ακμή με σκοπό την διαίρεση της ακμής, καθώς και η μετακίνηση και προσκόλληση μίας κορυφής πάνω σε συγκεκριμένη επιφάνεια. Αντίστοιχα για τον χειρισμό των επιφανειών παρέχονται επιλογές όπως η διαίρεση της επιφάνειας, η εναλλαγή εσωτερικής και εξωτερικής όψης και η αντιστροφή της σειράς των κορυφών που την απαρτίζουν. Επίσης είναι δυνατή η προσωρινή απόκρυψη μίας επιφάνειας ή ακόμα και ολόκληρης ομάδας για την διευκόλυνση της μοντελοποίησης.

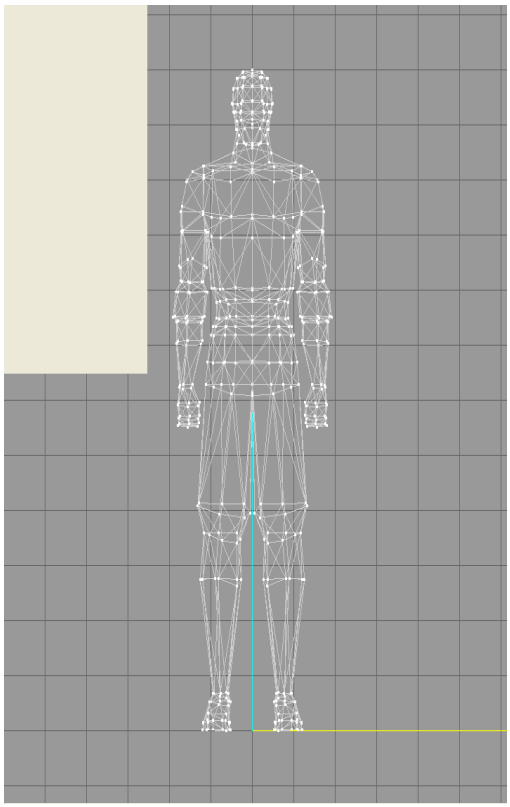
Για κάθε texture που αντιστοιχίζεται σε κάποια επιφάνεια του μοντέλου δημιουργείται στο MilkShape ένα νέο material. Στην περίπτωση των χαρακτήρων ο σχεδιασμός του texture που χρησιμοποιείται γίνεται με την βοήθεια του UVMapper. Ειδικού τύπου materials δημιουργούνται για τον καθορισμό του μεγέθους και της κλίμακας του αντικειμένου καθώς και τον προσδιορισμό των χαρακτηριστικών της κίνησης όταν πρόκειται για κινούμενο αντικείμενο. Στα materials που καθορίζουν το μέγεθος, την κλίμακα του αντικειμένου και τα γενικά χαρακτηριστικά της κίνησης αντιστοιχίζονται συγκεκριμένα ονόματα που αρχίζουν με το διακριτικό opt (options) . Για κάθε κίνηση του μοντέλου δημιουργείται ένα επιπλέον material με όνομα που ξεκινάει με το διακριτικό seq που υποδηλώνει ότι το συγκεκριμένο material αφορά σε μία ακολουθία (sequence) κίνησης. Μέσω αυτού του material καθορίζεται το είδος της κίνησης, τα frames που την αποτελούν, η ταχύτητα με την οποία θα εναλλάσσονται τα frames κατά την εκτέλεση της κίνησης καθώς και το αν η κίνηση θα είναι κυκλική, αν δηλαδή μετά το τέλος εκτέλεσης όλων των frames θα επαναεκτελούνται εκ νέου.

Όπως και στην περίπτωση των υπόλοιπων σχεδιαστικών προγραμμάτων που χρησιμοποιήθηκαν, το MilkShape επιτρέπει την ταυτόχρονη επισκόπηση του σχεδιαζόμενου αντικειμένου από διαφορετικές όψεις, μέσω πολλαπλών παραθύρων. Σε κάθε παράθυρο ο χρήστης μπορεί να επιλέξει την εμφάνιση μιας εκ των όψεων front, back, left, right, top, bottom και 3D.

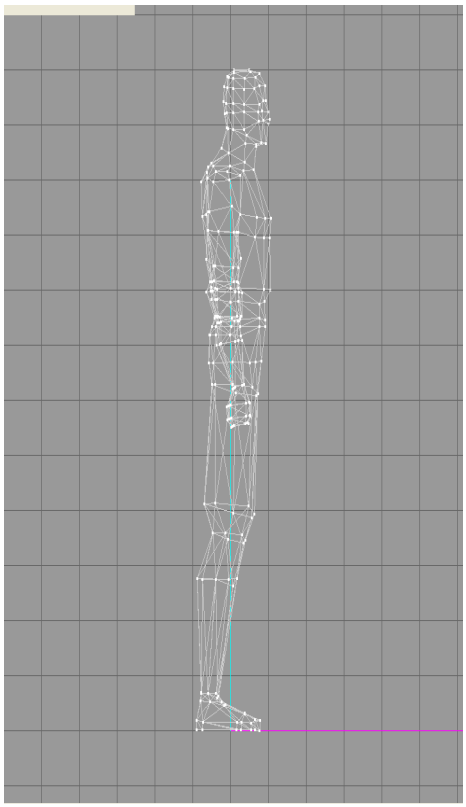
Εκτός από την μοντελοποίηση αντικειμένων το Milkshape παρέχει και την δυνατότητα σχεδιασμού της κίνησης των χαρακτήρων και των κινούμενων αντικειμένων. Συγκεκριμένα με την επιλογή Anim στο κάτω μέρος του παραθύρου της εφαρμογής διατίθεται στον χρήστη μία σειρά frames που μπορούν να χρησιμοποιηθούν για το σχεδιασμό της κίνησης. Με τις επιλογές του μενού Animate που αφορούν στην επιλογή keyframes, την αντιγραφή τους, την μετακίνησή τους και σε μία σειρά άλλες ενέργειες πραγματοποιείται ο σχεδιασμός της κίνησης. Τα εργαλεία επιλογής, κίνησης και περιστροφής που αναφέρθηκαν παραπάνω είναι απαραίτητα και σε αυτό το στάδιο. Παρόλο που το Milkshape παρέχει την δυνατότητα καθορισμού του animation ενός χαρακτήρα, για τον σκοπό αυτό χρησιμοποιήθηκέ το πρόγραμμα CaracterFX που περιγράφεται στο κεφάλαιο αυτό.

Τα αντικείμενα που σχεδιάζονται στο Milkshape αποθηκεύονται σε αρχεία τύπου ms3d. Για να είναι δυνατή η εισαγωγή και χρήση τους από το Torque είναι

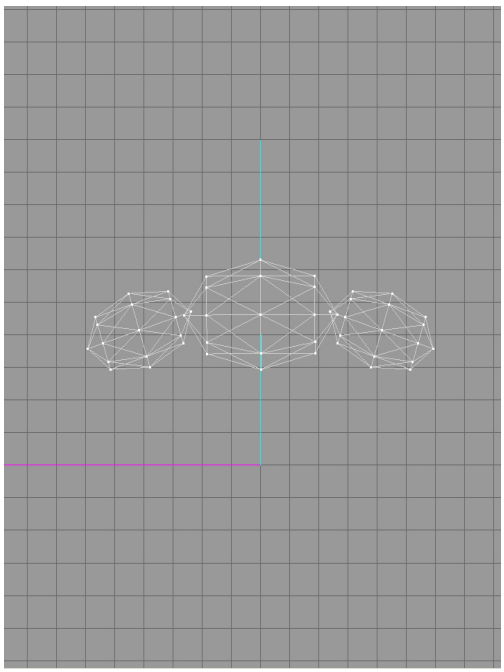
απαραίτητη η μετατροπή τους σε αρχεία τύπου dts. Για το σκοπό αυτό χρησιμοποιείται ο exporter Torque DTS Plus. Κατά την διαδικασία της εξαγωγής ενός αρχείου δίνεται η δυνατότητα καθορισμού ορισμένων χαρακτηριστικών του, όπως η κλίμακά του. Επίσης μπορούν να επιλεγούν τα textures που θα εξαχθούν και οι ακολουθίες κινήσεων. Η σχετική με την κίνηση πληροφορία μπορεί είτε να ενσωματωθεί στο αρχείο dts που θα προκύψει, ή να αποθηκευτεί σε ξεχωριστά αρχεία dsq. Αν επιλεγεί η δημιουργία αρχείων dsq κατασκευάζεται ένα αρχείο για κάθε ακολουθία κίνησης.



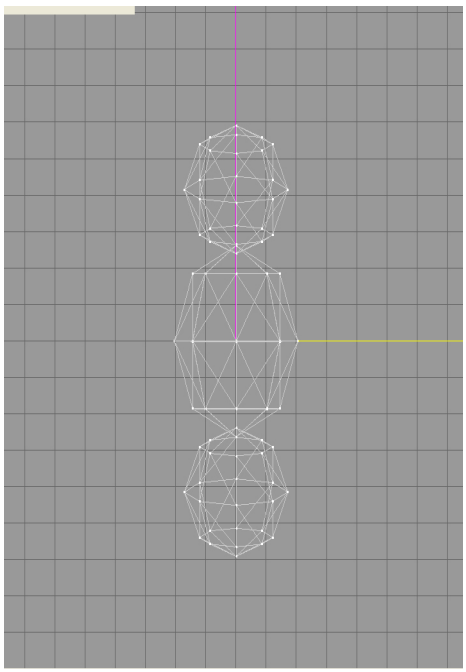
Εικόνα 2.9 Μοντελοποίηση χαρακτήρα στο Milkshape



Εικόνα 2.10 Μοντελοποίηση χαρακτήρα στο Milkshape



Εικόνα 2.11 Μοντελοποίηση αντικειμένου στο Milkshape



Εικόνα 2.12 Μοντελοποίηση αντικειμένου στο Milkshape

2.4 CharacterFX

Το πρόγραμμα CharacterFX χρησιμοποιήθηκε για τον σχεδιασμό της κίνησης των χαρακτήρων της εφαρμογής. Για την μεταφορά του χαρακτήρα από το MilkShape, όπου έγινε η μοντελοποίηση, στο CharacterFX απαιτείται η εξαγωγή του από το Milkshape σε μορφή αρχείου ASCII και η εισαγωγή του με την μορφή αυτή στο CharacterFX. Το CharacterFX περιέχει δύο χώρους εργασίας, το object mode και το animation mode.

Το object mode περιλαμβάνει εργαλεία κίνησης, περιστροφής και καθορισμού του μεγέθους πλεγμάτων και κόμβων. Επιπλέον μέσω του Material editor μπορεί να αντιστοιχηθεί σε κάθε πλέγμα του μοντέλου το κατάλληλο texture. Στο object mode πραγματοποιείται η διαδικασία κατασκευής του σκελετού του μοντέλου. Κατά την διαδικασία αυτή δημιουργείται μία ιεραρχία μεταξύ των κόμβων του σκελετού. Έτσι η κίνηση κόμβων που βρίσκονται σε χαμηλότερα επίπεδα της ιεραρχίας εξαρτάται άμεσα από την κίνηση των κόμβων που έχουν μεγαλύτερη ιεραρχική ισχύ. Μετά την κατασκευή του σκελετού πραγματοποιείται η διαδικασία αντιστοίχησης των κατάλληλων σημείων του πλέγματος στους κατάλληλους κόμβους του σκελετού. Το σύνολο των σημείων που αντιστοιχίζονται σε κάθε κόμβο του σκελετού ακολουθούν την κίνησή του, όταν αυτός μετακινείται. Η αντιστοίχηση αυτή πραγματοποιείται με χρήση κατάλληλων εργαλείων που παρέχει το πρόγραμμα για τον σκοπό αυτό.

Στο animation mode πραγματοποιείται ο σχεδιασμός των ακολουθιών της κίνησης. Στο παράθυρο του keyframer φαίνονται τα διαδοχικά frames που συνιστούν την κίνηση του χαρακτήρα. Ο σχεδιασμός της κίνησης πραγματοποιείται με καθορισμό της στάσης του χαρακτήρα σε κάθε frame του keyframer. Για τον σκοπό

αυτό χρησιμοποιούνται τα εργαλεία επιλογής, κίνησης και περιστροφής των κόμβων σκελετού. Με χρήση των εργαλείων αυτών, ο κάθε κόμβος τοποθετείται στην κατάλληλη θέση, ώστε να προκύψει η επιθυμητή στάση του σώματος του μοντέλου. Την κίνηση του κόμβου ακολουθούν όλα τα σημεία του μοντέλου που έχουν αντιστοιχηθεί στον συγκεκριμένο κόμβο ή σε κόμβους που εξαρτώνται από αυτόν. Στην πραγματικότητα η θέση των κόμβων του σκελετού καθορίζεται σε μερικά κομβικά frames, ενώ η θέση τους στα ενδιάμεσα frames καθορίζεται από το ίδιο το πρόγραμμα με χρήση μεθόδων παρεμβολής. Η μέθοδος παρεμβολής που χρησιμοποιείται καθώς και άλλα χαρακτηριστικά του σχεδιασμού της κίνησης, όπως ο ρυθμός εναλλαγής των frames κατά την εκτέλεση των διαφορετικών ακολουθιών κίνησης, μπορούν να καθοριστούν από τον χρήστη μέσω αντίστοιχων επιλογών. Ο keyframer περιλαμβάνει εργαλεία που διευκολύνουν τον σχεδιασμό του animation και αφορούν στην αντιγραφή και επικόλληση διαδοχικών frames, την μετακίνηση τους και την εκτέλεση επιπλέον βοηθητικών ενεργειών. Επίσης παρέχεται η δυνατότητα επισκόπησης της σχεδιαζόμενης κίνησης με την εκτέλεση συγκεκριμένων ακολουθιών διαδοχικών frames που ορίζει ο χρήστης.

Τα αρχεία που παράγονται από το πρόγραμμα CharacterFX είναι αρχεία τύπου cfx. Η απ' ευθείας εξαγωγή τους σε κάποια μορφή συμβατή με το Torque δεν είναι δυνατή. Γι' αυτό το λόγο είναι αναγκαία η εισαγωγή τους στο Milkshape σε μορφή ASCII. Με χρήση των exporter του Milkshape, παράγονται τα κατάλληλα αρχεία dts και dsq που χρησιμοποιούνται στη συνέχεια από το Torque.

2.5 Adobe Photoshop

Το πρόγραμμα Photoshop χρησιμοποιήθηκε για την επεξεργασία των textures που εμφανίζονται στον εικονικό κόσμο. Συγκεκριμένα με τη βοήθεια του προγράμματος αυτού επεξεργάστηκαν τα textures των αντικειμένων, των κτιρίων, του terrain και τα skins των χαρακτήρων.

Το Photoshop είναι ένα πρόγραμμα δημιουργίας και επεξεργασίας εικόνων. Περιλαμβάνει ένα πλήθος εργαλείων που χρησιμοποιούνται για την κατασκευή και τροποποίηση εικόνων. Τα εργαλεία αυτά αφορούν στην επιλογή και απομόνωση συγκεκριμένων τμημάτων της εικόνας, στην σχεδίαση γραμμών και των χρωματισμό περιοχών και στην επιλογή χρωματικών αποχρώσεων από τον κατάλογο χρωμάτων, από το χρωματικό φάσμα ή από συγκεκριμένες περιοχές της εικόνας. Επίσης το πρόγραμμα παρέχει εργαλεία που επιτρέπουν την εισαγωγή γεωμετρικών σχημάτων, την αντιγραφή περιοχών μίας εικόνας, την εισαγωγή κειμένου, την επιλογή γραμματοσειράς και μία σειρά άλλων επιλογών που συμβάλλουν στην διαμόρφωση της εικόνας.

Κατά την επεξεργασία της εικόνας υπάρχει η δυνατότητα δημιουργίας πολλαπλών επικαλυπτόμενων layers. Το καθένα από τα layers που απαρτίζουν μία εικόνα μπορούν να τροποποιηθούν ξεχωριστά και ανεξάρτητα από τα υπόλοιπα layers της ίδιας εικόνας.

Επιπλέον το πρόγραμμα παρέχει ένα σύνολο διαφορετικών φίλτρων που μπορούν να εφαρμοστούν σε μία εικόνα ή σε συγκεκριμένα τμήματα της. Η μορφή της εικόνας τροποποιείται με προσθήκη θορύβου ή με τροποποίηση χαρακτηριστικών της στοιχείων.

Το είδος της εικόνας και η ανάλυση της καθορίζεται από τον χρήστη. Έτσι όσον αφορά στο είδος της εικόνας ο χρήστης μπορεί να επιλέξει αν επιθυμεί να

κατασκευάσει μια εικόνα Bitmap, Grayscale, RGB Color, CMYC Color ή Lab Color. Αντίστοιχες επιλογές παρέχονται και για άλλα χαρακτηριστικά της εικόνας.

Τέλος το πρόγραμμα παρέχει εργαλεία για την μετακίνηση ή περιστροφή της εικόνας ή τμημάτων της, και την ρύθμιση χαρακτηριστικών όπως η φωτεινότητα και το contrast. Επίσης παρέχονται πολλές επιπλέον επιλογές και δυνατότητες που χρησιμοποιούνται για την διαμόρφωση της τελικής μορφής μίας εικόνας σύμφωνα με τις επιθυμίες του χρήστη.

3. ΟΡΓΑΝΩΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ ΣΥΜΦΩΝΑ ΜΕ ΤΗΝ ΑΡΧΙΤΕΚΤΟΝΙΚΉ ΠΕΛΑΤΗ\ΕΞΥΠΗΡΕΤΗΤΗ (CLIENT\SERVER).

Η εφαρμογή οργανώθηκε σύμφωνα με την αρχιτεκτονική πελάτη\ εξυπηρετητή (client\server architecture). Η οργάνωση αυτή διευκολύνει τον καταμερισμό εργασίας, την αποδοτικότερη διαχείριση των δεδομένων και την απόκρυψη πληροφοριών από το ένα μέρος στο άλλο όταν αυτό απαιτείται. Επίσης, η οργάνωση πελάτη\ εξυπηρετητή διευκολύνει την μελλοντική επέκταση της εφαρμογής ώστε να εξυπηρετεί περισσότερους πελάτες αν κάτι τέτοιο είναι επιθυμητό. Σε αυτήν την περίπτωση, ο κώδικας των πελατών και του εξυπηρετητή μπορούν να εκτελούνται σε διαφορετικούς υπολογιστές και να επικοινωνούν μέσω δικτύου.

Το τμήμα του πελάτη είναι υπεύθυνο κυρίως για την σύνδεση με τον εξυπηρετητή, τον έλεγχο της εισόδου και της εξόδου και την παρουσίαση της εικόνας. Το τμήμα του εξυπηρετητή είναι επιφορτισμένο με άλλου είδους αρμοδιότητες όπως η δημιουργία και διαγραφή αντικειμένων, η κίνηση των χαρακτήρων, η ανίχνευση σύγκρουσης μεταξύ αντικειμένων, η υλοποίηση των triggers και γενικότερα ο καθορισμός των περισσότερων συναρτήσεων και μεθόδων που αφορούν στην κατασκευή και την συμπεριφορά των αντικειμένων της εφαρμογής.

Η οργάνωση αυτή απαιτεί επιπλέον την υλοποίηση μεθόδων επικοινωνίας μεταξύ του τμήματος πελάτη και του τμήματος εξυπηρετητή. Οι κυριότεροι τρόποι υλοποίησης της επικοινωνίας είναι οι συναρτήσεις direct messaging, τα μηνύματα που στέλνονται από τον πελάτη στον εξυπηρετητή κατά την πραγματοποίηση συγκεκριμένων ενεργειών όπως η αλληλεπίδραση του χαρακτήρα με περιοχές που περικλείονται από triggers και η επικοινωνία μέσω του αντικειμένου της κλάσης GameConnection που υλοποιεί την σύνδεση. Οι τρόποι επικοινωνίας μεταξύ του πελάτη και του εξυπηρετητή περιγράφονται αναλυτικότερα στο τέλος του κεφαλαίου.

3.1 Ο common κώδικας της εφαρμογής

Δεδομένα, συναρτήσεις και μέθοδοι που χρησιμοποιούνται συχνά και καθορίζουν γενικά χαρακτηριστικά του τρόπου παρουσίασης των εφαρμογών και όχι χαρακτηριστικά της συγκεκριμένης εφαρμογής, βρίσκονται συγκεντρωμένα στον φάκελο common. Ο υπόλοιπος κώδικάς της εφαρμογής βρίσκεται αποθηκευμένος στον φάκελο Museum.

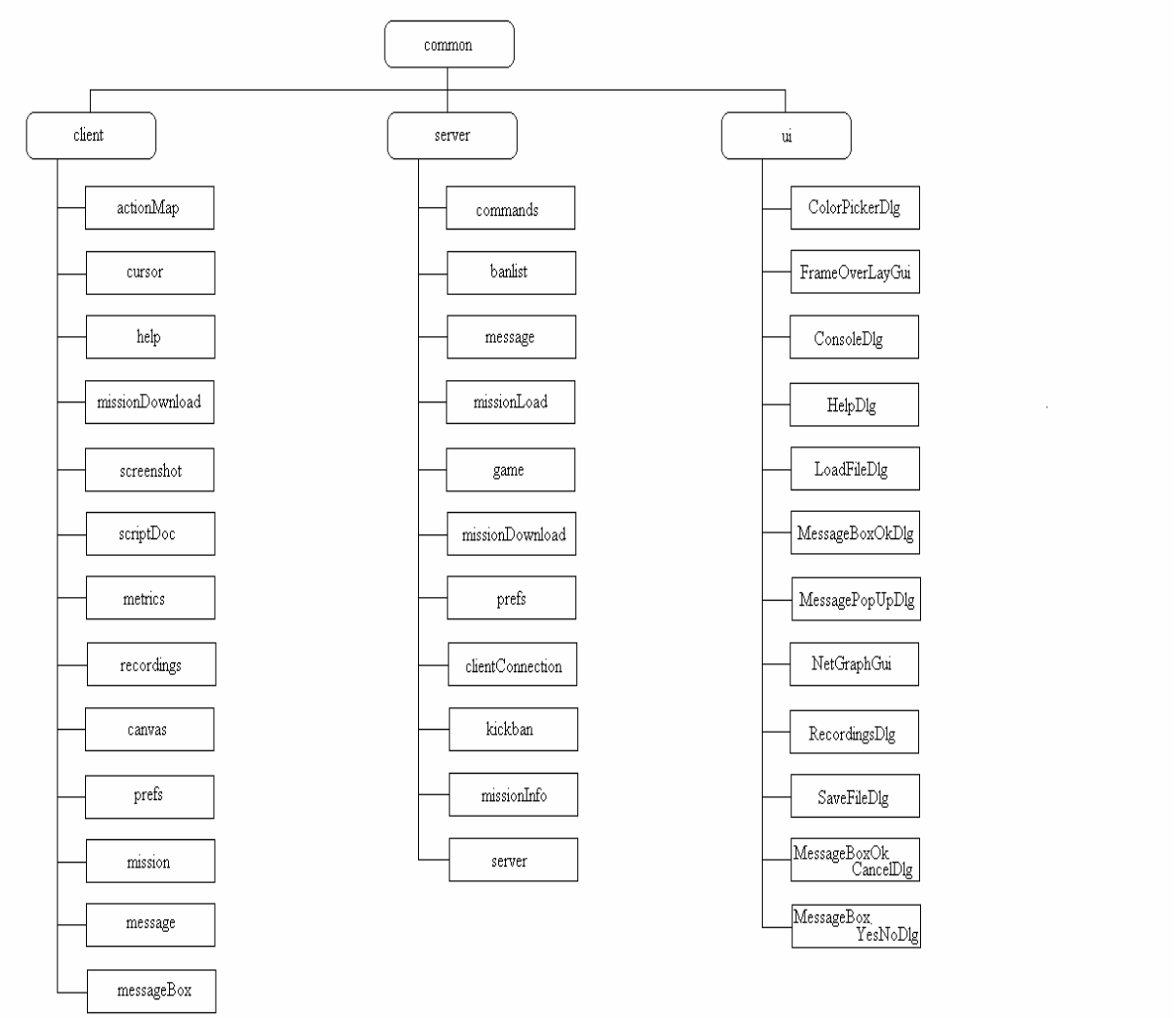
Ο φάκελος common περιλαμβάνει συναρτήσεις , μεθόδους και ορισμούς κλάσεων και datablock που καθορίζουν στοιχειώδεις γενικές λειτουργίες της εφαρμογής. Είναι οργανωμένος σε τρεις υποφακέλους με ονόματα server, client και ui. Η οργάνωση του φακέλου common φαίνεται στο σχήμα 3.1, οπού με ορθογώνια παραλληλόγραμμα αναπαριστάνονται τα αρχεία κώδικά με επέκταση .cs, ενώ με οβάλ οι υποφάκελοι του common.

Συγκεκριμένα ο common κώδικας που αφορά στον server περιλαμβάνει τα αρχεία Server, Commands, missionDownload, missionLoad, missionInfo, Message, Game, clientConnection και Prefs. Οι βασικές λειτουργίες που επιτελούνται σε αυτά τα αρχεία είναι η κατασκευή του server, η αρχικοποίησή του και η καταστροφή του

όταν αυτή απαιτείται, η δημιουργία σύνδεσης μεταξύ client και server και η αποσύνδεση όταν αυτή είναι επιθυμητή , ο καθορισμός των διάφορων φάσεων φόρτωσης της εφαρμογής, η υλοποίηση συναρτήσεων που διευκολύνουν την επικοινωνία μεταξύ client και server. Επίσης υλοποιείται ένα πλήθος συναρτήσεων και μεθόδων που καθορίζουν συμπληρωματικές ενέργειες που εκτελούνται κατά την διάρκεια των παραπάνω λειτουργιών καθώς και συναρτήσεις χειρισμού λαθών που αναλαμβάνουν την αποστολή διαγνωστικών μηνυμάτων όταν κάποια από τις λειτουργίες αποτύχει.

Ο κώδικας του φακέλου common που αφορά στον client περιλαμβάνει τα αρχεία actionMap, Canvas, Cursor, Help, Message, messageBox, Metrics, Mission, missionDownload, Prefs, recordings, screenshot και scriptDoc. Οι κύριες λειτουργίες που υλοποιεί ο κώδικας των παραπάνω αρχείων είναι η εμφάνιση του παραθύρου της εφαρμογής στην πλευρά του client, η χρήση του κέρσορα και καθορισμός συναρτήσεων που χρησιμεύουν στο binding, ο καθορισμός χαρακτηριστικών του παραθύρου βοήθειας, η αναπροσαρμογή του μεγέθους των παραθύρων διαλόγου ανάλογα με το μήκος του μηνύματος που φέρουν, η υλοποίηση συναρτήσεων σχετικών με την αναπαραγωγή του video της εφαρμογής, και αποθήκευση στιγμιότύπων του. Επίσης υλοποιούνται συναρτήσεις που έχουν να κάνουν με την επικοινωνία μεταξύ client και server και την ανταπόκριση του client σε μηνύματα του server που αφορούν στην έναρξη και τερματισμό της εφαρμογής.

Ο κώδικας του φακέλου common που αφορά στα ui περιέχει τα αρχεία ColorPickerDlg, ConsoleDlg, FrameOverlayGui, HelpDlg, LoadFileDlg, MessageBoxOKCancelDlg, MessageBoxOkDlg, MessageBoxYesNoDlg, MessagePopupDlg, NetGraphGui, RecordingsDlg και SaveFileDlg. Τα αρχεία αυτά περιέχουν τον κώδικα που καθορίζει τα γενικά χαρακτηριστικά των κυριότερων παραθύρων διαλόγου, του παραθύρου της κονσόλας και ορισμένων ακόμα συστατικών στοιχείων των GUIs.



Σχήμα3.1 Οργάνωση του φακέλου common.

3.2 Ο κώδικάς του Museum

Λόγω της οργάνωσης της εφαρμογής σύμφωνα με την αρχιτεκτονική client\server, ο κώδικάς του φακέλου Museum χωρίζεται στο τμήμα του πελάτη (client), στο τμήμα του εξυπηρετητή (server) και στο τμήμα των δεδομένων (data). Η επιλογή της συγκεκριμένης δομής, όπως αναφέρθηκε παραπάνω, έγινε γιατί επιτρέπει την δυναμική προσθήκη περισσότερων του ενός επισκεπτών αν αυτό είναι επιθυμητό και επίσης μπορεί να κάνει αδιαφανή στον χρήστη κάποια κομμάτια του κώδικα του εξυπηρετητή. Έτσι το τμήμα του κώδικα που υλοποιεί το μεγαλύτερο κομμάτι της εφαρμογής βρίσκεται στον φάκελο του server, ενώ ο φάκελος του client περιλαμβάνει μόνο τον κώδικα που αφορά στην επικοινωνία του συγκεκριμένου επισκέπτη με την εφαρμογή. Ο φάκελος των δεδομένων περιέχει τα απαραίτητα δεδομένα για την υλοποίηση του εικονικού κόσμου. Η οργάνωση του φακέλου Museum φαίνεται στο σχήμα 3.2, οπού με ορθογώνια παραλληλόγραμμα αναπαριστάνονται τα αρχεία

κώδικά με επέκταση .cs, ενώ με οβάλ οι υποφάκελοι του Museum.

3.2.1 Το τμήμα του εξυπηρετητή

Συγκεκριμένα το τμήμα του εξυπηρετητή (server) περιέχει τα αρχεία camera, editor, visitor, museumEmp, items, guide, triggers, markers και museumCode. Τα αρχεία αυτά περιέχουν τον κώδικα που είναι υπεύθυνος για τις λειτουργίες που αναφέρονται συνοπτικά παρακάτω και επεξήγονται αναλυτικότερα στα επόμενα κεφάλαια.
camera. Στο αρχείο αυτό ορίζεται το datablock για την κατασκευή του αντικειμένου της κάμερας όταν είναι σε observer mode και καθορίζεται η default τιμή της ταχύτητάς της.
editor. Το αρχείο αυτό περιλαμβάνει δηλώσεις datablock για τύπους αντικειμένων που χρησιμοποιούνται κατά την κατασκευή του εικονικού κόσμου. Επίσης ορίζονται οι μέθοδοι create των datablock αυτών, που καλούνται από τον world editor και ορισμένες συναρτήσεις χειρισμού της κάμερας, που χρησιμοποιεί ο editor.
visitor. Περιέχει το datablock TSShapeConstructor το οποίο καθορίζει την μορφή του επισκέπτη και τις ακολουθίες κινήσεών του, καθώς και το datablock του επισκέπτη και τις μεθόδους του.
museumEmp. Περιέχει το datablock που ορίζει την μορφή του υπαλλήλου του μουσείου που βρίσκεται στο γραφείο πληροφοριών.
items. Ορίζονται τα datablock των αντικειμένων του εικονικού κόσμου τα οποία αλληλεπιδρούν με τους χαρακτήρες της εφαρμογής. Επίσης ορίζονται οι συναρτήσεις και μέθοδοι που υλοποιούν την αλληλεπίδραση αυτή.
guide. Στο αρχείο αυτό περιέχονται όλες οι μέθοδοι των αντικειμένων των κλάσεων AIManager και AIPlayer και συναρτήσεις που δίνουν εντολή για την δημιουργία αντικείμένων των κλάσεων αυτών. Οι μέθοδοι των αντικειμένων AIManager αναλαμβάνουν την κατασκευή στιγμιοτύπων της κλάσης AIPlayer, ενώ οι μέθοδοι των αντικειμένων AIPlayer αναλαμβάνουν την κίνηση και τις υπόλοιπες ενέργειες που μπορεί να εκτελέσει ένας χαρακτήρας που ανήκει σε αυτή τη κλάση, όπως ο ξεναγός του μουσείου.
triggers. Στο αρχείο αυτό ορίζεται ένα datablock για κάθε είδους trigger που χρησιμοποιείται στην εφαρμογή. Επίσης ορίζονται οι μέθοδοι που καθορίζουν την συμπεριφορά κάθε trigger κατά την είσοδο, την έξοδο ή την παραμονή κάποιου χαρακτήρα στον χώρο του trigger.
markers. Περιέχει τα datablock που ορίζουν τις ιδιότητες των αντικειμένων τύπου WayPointMarker και SpawnSphereMarker, καθώς και έναν κατασκευαστή αυτών.
museumCode. Αρχικοποιεί τον server, πραγματοποιώντας την φόρτωση του κώδικα όλων των παραπάνω αρχείων κατά την έναρξη της εφαρμογής και καταστρέφει την σύνδεση στο τέλος της εφαρμογής. Επίσης υλοποιεί συναρτήσεις και μεθόδους που αφορούν στις επιπλέον ενέργειες που πρέπει να εκτελούνται κατά την φόρτωση και λήξη εκτέλεσης της εφαρμογής, την κατασκευή του χαρακτήρα του επισκέπτη και την είσοδο και έξοδό του από τον εικονικό κόσμο.

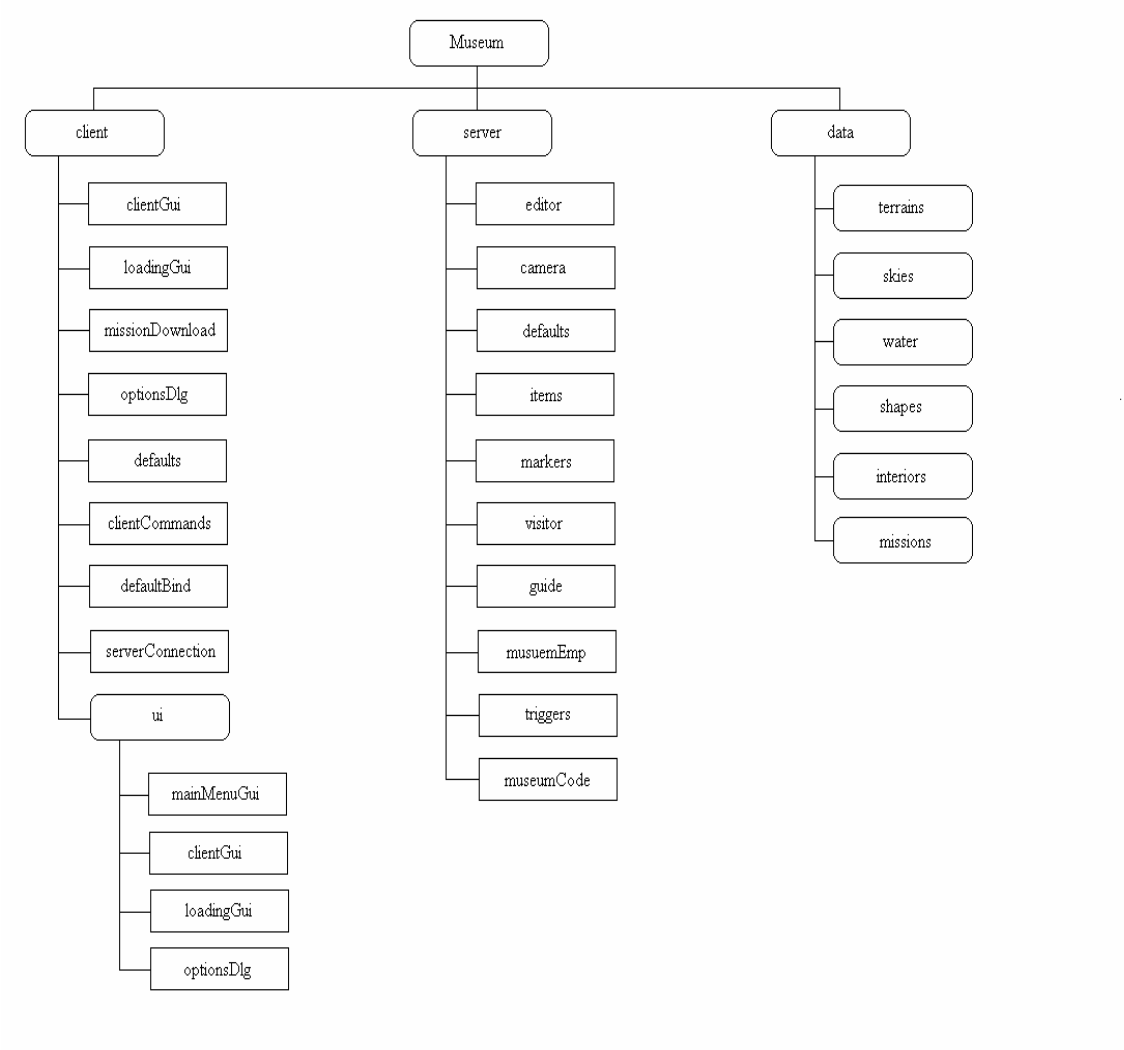
3.2.2 Το τμήμα του πελάτη

Το τμήμα του πελάτη(client), περιέχει τα αρχεία clientGui, loadingGui, serverConnection, defaultBind, missionDownload, clientCommands, defaults,

optionsDlg, καθώς και τα περιεχόμενα του φακέλου ui τα οποία περιγράφονται αναλυτικά στο κεφάλαιο που αφορά στην επικοινωνία χρήστη εφαρμογής μέσω GUIs. Ο κώδικας των παραπάνω αρχείων αναλαμβάνει την εκτέλεση των εξής ενεργειών.
clientGui. Στο αρχείο αυτό υλοποιούνται οι μέθοδοι onWake και onSleep του αντικειμένου playGui που αποτελεί το βασικότερο στιγμιότυπο της κλάσης TSControl, μέσω του οποίου παρουσιάζεται στην οθόνη η εφαρμογή. Το αντικείμενο αυτό ορίζεται στο αρχείο Museum\client\ui\clientGui
loadingGui. Στο αρχείο αυτό υλοποιούνται οι μέθοδοι onAdd, onWake και onSleep του αντικειμένου LoadingGui τύπου GuiChunkedBitmapCtrl που ορίζεται στο αρχείο Museum\client\ui\loadingGui.
serverConnection. Περιλαμβάνει συναρτήσεις και μεθόδους που καλούνται κατά την διαδικασία δημιουργίας σύνδεσης μεταξύ client και server και αντιμετωπίζουν καταστάσεις που προκύπτουν κατά την διαδικασία αυτή, καθώς και κατά την διαδικασία αποσύνδεσης.
defaultBind. Καθορίζεται το binding, δηλαδή αντιστοιχίζονται συγκεκριμένες συναρτήσεις ή μέθοδοι σε πλήκτρα του πληκτρολογίου και σε κινήσεις του ποντικιού. Με αυτόν τον τρόπο, πάτημα τον συγκεκριμένων πλήκτρων ή οι κινήσεις του ποντικιού προκαλούν την εκτέλεση προκαθορισμένων ενεργειών.
missionDownload. Στο αρχείο αυτό υλοποιούνται οι συναρτήσεις και μέθοδοι που καλείται να εκτελέσει ο client κατά την διάρκεια φόρτωσης της εφαρμογής καθώς και συναρτήσεις και μέθοδοι ανταλλαγής μηνυμάτων που ενημερώνουν για την εξέλιξη της φόρτωσης.
clientCommands. Υλοποιούνται οι συναρτήσεις που καλούνται από τον client όταν αυτός λάβει τα αντίστοιχα μηνύματα από τον server με την διαδικασία του direct massaging που περιγράφεται παρακάτω.
defaults. Περιέχει τις default τιμές βασικών παραμέτρων της εφαρμογής.
optionsDlg. Καθορίζει τις ενέργειες που εκτελούνται σύμφωνα με τις επιλογές που κάνει ο χρήστης, μέσω του παραθύρου options του main menu. Το παράθυρο αυτό δίνει την δυνατότητα στον χρήστη να ρυθμίσει κάποια χαρακτηριστικά που αφορούν στην παρουσίαση της εφαρμογής.

3.2.3 Το τμήμα των δεδομένων

Ο φάκελος data περιλαμβάνει τον κώδικα κατασκευής του περιβάλλοντος που βρίσκεται στο αρχείο MuseumMission του υποφακέλου missions. Ο data περιλαμβάνει επίσης τους υποφακέλους interiors, shapes, terrains, skies και water. Καθένας από αυτούς τους φακέλους περιέχει πληροφορίες για τα αντικείμενα που ανήκουν σε κάθε μία από αυτές τις κατηγορίες. Συγκεκριμένα για κάθε αντικείμενο υπάρχει ένα αρχείο που περιέχει πληροφορίες για την μορφή του και όλα τα textures που χρησιμοποιεί.



Σχήμα3.2 Οργάνωση του φακέλου Museum.

3.3 Επικοινωνία ανάμεσα στον πελάτη και τον εξυπηρετητή

Υπάρχουν διάφοροι τρόποι επικοινωνίας μεταξύ του client και του server. Οι τρόποι που χρησιμοποιούνται στην εφαρμογή περιγράφονται παρακάτω.

3.3.1 Επικοινωνία μέσω συναρτήσεων απ' ευθείας ανταλλαγής μηνυμάτων (direct messaging functions).

Οι συναρτήσεις αυτές είναι οι CommandToServer και CommandToClient. Η συνάρτηση CommandToServer χρησιμοποιείται για αποστολή μηνύματος από τον client στον server. Δέχεται σαν ορίσματα την συνάρτηση που ζητάει ο client από τον server να εκτελέσει και τα ορίσματά αυτής. Η συνάρτηση-όρισμα της CommandToServer υλοποιείται στον κώδικα του server και το όνομα της φέρει το πρόθεμα ServerCmd. Αντίστοιχα όταν ο server επιθυμεί να εκτελέσει ο client κάποια συνάρτηση στέλνει ένα μήνυμα CommandToClient. Ο τρόπος που είναι οργανωμένη

η εφαρμογή υποστηρίζει την εισαγωγή περισσότερων του ενός client, αν αυτό είναι επιθυμητό. Έτσι η συνάρτηση CommandToClient περιέχει ως ορίσματα, την συνάρτηση που ζητάει ο server από τον client να εκτελέσει, τα ορίσματα αυτής και επιπλέον τον handle του client στον οποίο αποστέλεται το μήνυμα. Ο κώδικας της συνάρτησης-ορίσματος της CommandToClient βρίσκεται στο τμήμα του client και φέρει το πρόθεμα ClientCmd.

Ένα παράδειγμα χρήσης των συναρτήσεων direct messaging είναι η περίπτωση που ο επισκέπτης πλησιάζει κάποιο από τα εκθέματα του μουσείου. Σε αυτή την περίπτωση, καλείται η συνάρτηση commandToClient με ορίσματα τον handle του client που πλησίασε το έκθεμα, την συνάρτηση ShowObjectInfo και τον αριθμό του εκθέματος. Στον κώδικα του client υπάρχει η συνάρτηση clientCmdShowObjectInfo που δέχεται σαν όρισμα τον αριθμό του εκθέματος και εμφανίζει τις σχετικές πληροφορίες στην οθόνη του χρήστη.

3.3.2 Επικοινωνία μέσω χωρικών triggers(area triggers).

Τα χωρικά triggers είναι αντικείμενα που οριοθετούν μια περιοχή του εικονικού κόσμου. Στην συνέχεια με υλοποίηση των κατάλληλων συναρτήσεων και μεθόδων προσδίδεται συγκεκριμένη συμπεριφορά στην περιοχή αυτή. Έτσι, όταν κάποιο αντικείμενο ή χαρακτήρας εισέρχεται, εξέρχεται ή παραμένει στην οριοθετημένη από το trigger περιοχή στέλνεται στον server ένα μήνυμα και αυτός καλεί την συνάρτηση που διαχειρίζεται το γεγονός. Για παράδειγμα όταν ο επισκέπτης εισέρχεται στην έκθεση ζωγραφικής του μουσείου που εμπεριέχεται στο trigger ExpoTwoTrigger στέλνεται ένα μήνυμα στον server και καλείται η μέθοδος onEnterTrigger του trigger αυτού και εμφανίζονται τα κατάλληλα μηνύματα πληροφοριών στην οθόνη του χρήστη.

3.3.3 Επικοινωνία μέσω των συναρτήσεων του αντικειμένου GameConnection.

Κατά την δημιουργία της σύνδεσης μεταξύ client και server κατασκευάζεται ένα στιγμιότυπο της κλάσης GameConnection. Η κλάση GameConnection περιλαμβάνει μεθόδους μέσω τον οποίων είναι δυνατή η ανταλλαγή μηνυμάτων μεταξύ client και server. Οι μέθοδοι αυτές χρησιμοποιούνται κυρίως για την επικοινωνία στο στάδιο της σύνδεσης και της φόρτωσης της εφαρμογής. Παραδείγματα τέτοιων μεθόδων είναι οι onConnectionAccepted και onConnectionError που καλούνται όταν πραγματοποιείται με επιτυχία ή σύνδεση και όταν η σύνδεση αποτυγχάνει, αντίστοιχα.

4. ΣΧΕΔΙΑΣΜΟΣ ΤΟΥ ΕΙΚΟΝΙΚΟΥ ΚΟΣΜΟΥ.

Η διαμόρφωση του περιβάλλοντος του εικονικού κόσμου της εφαρμογής πραγματοποιείται με την βοήθεια του world editor του Torque game engine. Ο world editor εμφανίζει το περιβάλλον του εικονικού κόσμου με όλα τα συστατικά του αντικείμενα. Τα αντικείμενα αυτά είναι οργανωμένα σε μία δενδρική δομή που επιτρέπει την επιλογή οποιουδήποτε από αυτά και την εμφάνιση των επιμέρους χαρακτηριστικών του. Επίσης ο σχεδιαστής του εικονικού κόσμου έχει την δυνατότητα μέσω του world editor να εισάγει στον εικονικό κόσμο νέα αντικείμενα να τα τοποθετήσει στις θέσεις που επιθυμεί και να καθορίζει τα χαρακτηριστικά τους. Έτσι ο world editor αποτελεί ένα εύχρηστο interface για την δημιουργία και τροποποίηση του εικονικού κόσμου και παρέχει ένα οπτικό αποτέλεσμα του περιβάλλοντος που καθιστά ευκολότερη την επεξεργασία του, με την άμεση επισκόπηση των αλλαγών που πραγματοποιούνται. Ο κώδικας κατασκευής όλων των συστατικών του εικονικού κόσμου βρίσκεται αποθηκευμένος στο αρχείο Museum\data\missions\MuseumMission. Επομένως, οποιαδήποτε αλλαγή σε χαρακτηριστικά υπαρχόντων αντικειμένων ή προσθήκη νέων αντικειμένων μπορεί να πραγματοποιηθεί με τροποποίηση του κώδικα αυτού του αρχείου. Με τον τρόπο αυτό όμως δεν είναι δυνατή η ταυτόχρονη επισκόπηση του αποτελέσματος που προκύπτει.

Για την κατασκευή του εικονικού κόσμου χρησιμοποιείται πλήθος αντικειμένων που κατασκευάζονται με την βοήθεια σχεδιαστικών προγραμμάτων και στη συνέχεια εισάγονται στο Torque game engine. Τα προγράμματα που χρησιμοποιήθηκαν για τον σκοπό αυτό είναι τα Quake Army Knife (QuArK) και MilkShape. Τα αρχεία που εισάγονται στο Torque από το QuArK έχουν την επέκταση dif και ανήκουν στην κλάση InteriorInstance, ενώ τα αρχεία που εισάγονται από το MilkShape έχουν την επέκταση dts και ανήκουν στις κλάσεις TSStatic και StaticShape. Τα texture των αντικειμένων, του terrain και οι εικόνες που χρησιμοποιήθηκαν για την κατασκευή του ουρανού και των σύννεφων δημιουργήθηκαν και επεξεργάστηκαν στο Adobe Photoshop και αποτελούν εικόνες τύπου jpeg και png.

Τα κύρια συστατικά στοιχεία του εικονικού κόσμου της εφαρμογής είναι το περιβάλλον του εξωτερικού χώρου, τα κτίρια και τα αντικείμενα. Επίσης στον εικονικό κόσμο συμπεριλαμβάνεται μία σειρά αντικειμένων που δεν είναι ορατά κατά την διάρκεια εκτέλεσης της εφαρμογής, αλλά παίζουν σημαντικό ρόλο στον έλεγχο συγκεκριμένων δραστηριοτήτων και στον καθορισμό της ροής εκτέλεσης της εφαρμογής. Τέλος υπάρχουν αντικείμενα που συμβάλουν στην καλύτερη οργάνωση των συστατικών του περιβάλλοντος και διευκολύνουν τη σχεδίαση του εικονικού κόσμου.

Ακολουθεί μια λεπτομερέστερη περιγραφή των στοιχείων που απαρτίζουν τον εικονικό κόσμο.

4.1 Δημιουργία περιβάλλοντος εξωτερικών χώρων

Πρόκειται για το σύνολο των αντικειμένων που δίνουν στον χρήστη της εφαρμογής την αίσθηση του φυσικού κόσμου όταν ο επισκέπτης του εικονικού κόσμου βρίσκεται στον εξωτερικό χώρο του περιβάλλοντος. Τα κύρια συστατικά του περιγράφονται παρακάτω.

4.1.1 Ο χώρος που περιλαμβάνει το περιβάλλον της εφαρμογής.

Πρόκειται για ένα αντικείμενο της κλάσης MissionArea που καθορίζει τα όρια της περιοχής την οποία μπορεί να καταλαμβάνει ο εικονικός κόσμος.

4.1.2 Ο ουρανός.

Αποτελεί στιγμιότυπο της κλάσης Sky. Πρόκειται για ένα παραλληλεπίπεδο μεγάλων διαστάσεων με διαφορετικά textures να καλύπτουν τις έξι έδρες του. Οι έδρες του διατηρούν πάντα σταθερή απόσταση από τον επισκέπτη και επομένως ο επισκέπτης δεν μπορεί ποτέ να φτάσει τα όρια του παραλληλεπιπέδου. Με αυτόν τον τρόπο του δημιουργείται η ψευδαίσθηση ότι ο ουρανός βρίσκεται σε άπειρη απόσταση και έτσι δεν αντιλαμβάνεται ότι η αναπαράστασή του γίνεται με την βοήθεια του παραλληλεπιπέδου. Η θέση του, η κλήση του ως προς κάθε άξονα και η κλίμακα του καθορίζονται από τις τιμές των παραμέτρων position, rotation και scale αντίστοιχα. Το αρχείο Museum/data/skies/sky_day.dml που αποτελεί τιμή της παραμέτρου materialList περιέχει τα ονόματα των εικόνων που θα καλύψουν κάθε έδρα του παραλληλεπιπέδου και των εικόνων που θα χρησιμοποιηθούν για την παρουσίαση των σύννεφων. Το texture της κάτω πλευράς του παραλληλεπιπέδου δεν είναι ορατό, όπως υποδηλώνει η τιμή 0 της παραμέτρου renderBottomTexture. Υπάρχει δυνατότητα ο ουρανός να μην παρουσιαστεί σύμφωνα με το μοντέλο του παραλληλεπιπέδου, αν η τιμή της μεταβλητής useSkyTextures μετατραπεί από 1 σε 0. Στην περίπτωση αυτή ο ουρανός καλύπτεται από το χρώμα που ορίζει η παράμετρος SkySolidColor.

Καθορίζονται τρία επίπεδα συννεφων σε ύψη που ορίζονται από τις τιμές των παραμέτρων cloudHeightPer[0], cloudHeightPer[1] και cloudHeightPer[2]. Σύμφωνα με την παρατήρηση οτι η ταχύτητα των συννεφων αυξάνεται στα ψηλότερα στρώματα της ατμόσφαιρας ορίζονται διαφορετικές ταχύτητες σε κάθε στρώμα που ορίζονται από τις τιμές των παραμέτρων cloudSpeed1, cloudSpeed2 και cloudSpeed3.

Η ομίχλη μπορεί και αυτή να υλοποιηθεί σε επίπεδα διαφορετικής πυκνότητας. Στον εικονικό κόσμο της εφαρμογής χρησιμοποιείται το ένα μόνο από τα διαθέσιμα επίπεδα. Η έκταση του χώρου που θα καταλαμβάνει η ομίχλη καθορίζεται από την τιμή της παραμέτρου fogVolume1, ενώ το χρώμα της από την τιμή της παραμέτρου fogVolumeColor1. Εκτός από τα επίπεδα αυτά υπάρχει και η ομίχλη που εμφανίζεται πάντα σε συγκεκριμένη απόσταση από τον επισκέπτη και καθιστά μη ορατά τα αντικείμενα που βρίσκονται σε μεγαλύτερες αποστάσεις. Η απόσταση αυτή καθορίζεται από την τιμή της παραμέτρου fogDistance και το χρώμα της από την παράμετρο fogColor.

Οι παράμετροι windVelocity και windEffectPrecipitation καθορίζουν τα χαρακτηριστικά του ανέμου και βρίσκουν εφαρμογή στην περίπτωση εισαγωγής φαινομένων καταιγίδας στον εικονικό κόσμο.

Εικόνα 4.1 Ο ουρανός

4.1.3 Ο ήλιος.

Πρόκειται για στιγμιότυπο της κλάσης Sun. Το αντικείμενο που αναπαριστά τον ήλιο δεν είναι ορατό στον world editor. Σκοπός του είναι να καθορίσει τον τρόπο που φωτίζονται τα αντικείμενα που απαρτίζουν το περιβάλλον της εφαρμογής. Μέσω του world editor μπορεί να καθοριστεί η θέση του αντικειμένου που αναπαριστά τον ήλιο, η κλήση του ως προς κάθε άξονα και η κλίμακα του καθώς και το χρώμα, η ένταση και η κατεύθυνση του φωτός που εκπέμπει.

Σύμφωνα με την θέση του ήλιου δημιουργούνται οι σκιές των αντικειμένων. Επίσης οι πλευρές των αντικειμένων που δεν βρίσκονται προς την πλευρά του ήλιου είναι πιο σκουρόχρωμες. Το χρώμα και η ένταση του φωτός που επηρεάζει τα αντικείμενα και δημιουργεί τις σκιές τους καθορίζεται από την τιμή της παραμέτρου color. Η τιμή της παραμέτρου ambient καθορίζει το χρώμα και την ένταση του φωτός που υπάρχει διάχυτο στο περιβάλλον της εφαρμογής.

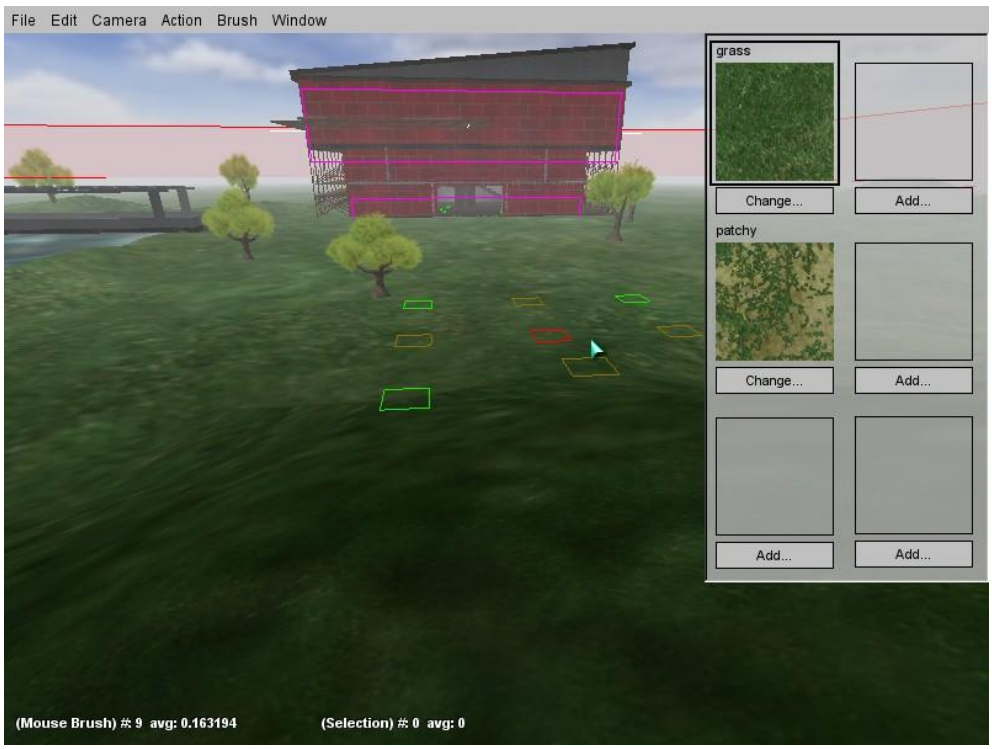
Εικόνα 4.2 Ο ήλιος

4.1.4 Το έδαφος.

Δυο είναι τα κύρια χαρακτηριστικά που καθορίζουν την εμφάνιση του εδάφους: η μορφή του σε τοπογραφικό επίπεδο και το texture που το καλύπτει. Τα τοπογραφικά χαρακτηριστικά του εδάφους μπορούν να διαμορφωθούν με την βοήθεια του terrain editor του world editor. Ο terrain editor περιλαμβάνει το εργαλείο brush με την βοήθεια του οποίου μπορούν να δημιουργηθούν υψώματα στα επιθυμητά σημεία του terrain. Για την επεξεργασία του terrain μπορεί επίσης να χρησιμοποιηθεί το εργαλείο terrain terraform editor του world editor.

Το texture που καλύπτει το έδαφος της εφαρμογής είναι μία 256×256 εικόνα png το περιεχόμενο της οποίας επαναλαμβάνεται σε όλη την επιφάνεια του εδάφους του περιβάλλοντος. Ο τρόπος με τον οποίο εφαρμόζεται το texture στην επιφάνεια του terrain μπορεί να καθοριστεί με την βοήθεια του terrain texture editor. Διαφορετικά texture μπορούν να εφαρμοστούν σε ορισμένα σημεία του terrain με την βοήθεια του terrain texture painter. Το terrain αποτελεί στιγμιότυπο της κλάσης TerrainBlock και το texture του καθορίζεται από την παράμετρο detailTexture. Το texture που καλύπτει το terrain είναι η εικόνα Museum/data/terrains/grass.jpg ενώ οι περισσότερες λεπτομέρειες για την μορφή του βρίσκονται στο αρχείο Museum/data/missions/MuseumMission.ter.

Στο εξωτερικό περιβάλλον προστέθηκαν επίσης δέντρα που εισάγονται στο περιβάλλον ως αντικείμενα τύπου dts. Τα αρχεία αυτά και τα απαραίτητα texture που απαιτούνται για τον κορμό και το φύλλωμα των δέντρων βρίσκονται στο φάκελο Museum\data\shapes\trees.



Εικόνα 4.3 Κατασκευή του terrain.

4.1.5 Η λίμνη.

Πρόκειται για το στιγμιότυπο της κλάσης WaterBlock με το όνομα lake. Κατά την κατασκευή του αντικειμένου καθορίζονται πολλά από τα χαρακτηριστικά που διαμορφώνουν την μορφή του. Οι τιμές των παραμέτρων surfaceTexture και ShoreTexture ορίζουν τα texture που χρησιμοποιούνται για την επιφάνεια του νερού και των ρηχών περιοχών αντίστοιχα, ενώ οι τιμές των παραμέτρων envMapOverTexture και envMapUnderTexture καθορίζουν την όψή της επιφάνειας του νερού για έναν παρατηρητή που την κοιτάζει από πάνω ή από τον βυθό αντίστοιχα. Σε όλες αυτές τις μεταβλητές έχει δοθεί ως τιμή η εικόνα που βρίσκεται αποθηκευμένη στο αρχείο Museum /data/water/waterblack.jpg. Το βάθος από το οποίο θα ξεκινάει η εφαρμογή του texture που αντιστοιχεί σε ρηχή περιοχή καθορίζεται από την παράμετρο ShoreDepth. Οι παράμετροι MinAlpha και MaxAlpha ορίζουν την ελάχιστη και μέγιστη απόσταση του νερού από το έδαφος ενώ η τιμή 1 της παραμέτρου DepthGradient υποδηλώνει ότι μεταξύ ρηχών και βαθύτερων περιοχών θα πραγματοποιείται γραμμική παρεμβολή των αντίστοιχων texture.

Ο βαθμός διαφάνειας του νερού καθορίζεται από την παράμετρο surfaceOpacity και η πυκνότητά του από την παράμετρο density. Οι τιμές 0 των παραμέτρων FlowAngle και FlowRate υποδηλώνουν ότι δεν θα υπάρχει ροή νερού προς καμία κατεύθυνση. Τα χαρακτηριστικά της διαταραχής στην επιφάνεια του νερού καθορίζονται από τις τιμές των παραμέτρων DistortGridScale, DistortMag και DistortTime. Οι μεταβλητές TessSurface και TessShore καθορίζουν πόσες φόρες θα επαναλαμβάνονται τα αντίστοιχα texture σε κάθε μπλοκ της επιφάνειας του νερού.

Τέλος υπάρχει ένα πλήθος παραμέτρων που καθορίζουν το μέγεθος, την κλήση,

την κλίμάκα της λίμνης καθώς και την ανακλαστικότητα της επιφάνειάς της, του texture που θα αντικατοπτρίζεται και τα υπόλοιπα χαρακτηριστικά που συμβάλλουν στο οπτικό αποτέλεσμα της λίμνης.

Εικόνα 4.4 Η λίμνη

Ακολουθεί το τμήμα του κώδικα του αρχείου Museum\data\missions\MuseumMission που κατασκευάζει το εξωτερικό περιβάλλον του εικονικού κόσμου.

```
//--------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\data\missions\MuseumMission
//--------------------------------------------------------------------

new MissionArea(MissionArea) {
      Area = "-360 -648 720 1296";
      flightCeiling = "300";
      flightCeilingRange = "20";
         locked = "true";
  };

new Sky(Sky) {
```

```
      position = "336 136 0";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      materialList = "~/data/skies/sky_day.dml";
      cloudHeightPer[0] = "0.3";
      cloudHeightPer[1] = "0.2";
      cloudHeightPer[2] = "0.1";
      cloudSpeed1 = "0.0004";
      cloudSpeed2 = "0.0006";
      cloudSpeed3 = "0.0008";
      fogDistance = "500";
      fogColor = "0.82 0.828 0.844 1";
      fogStorm1 = "0";
      fogStorm2 = "0";
      fogStorm3 = "0";
      fogVolume1 = "300 0 20";
      fogVolume2 = "0 0 0";
      fogVolume3 = "0 0 0";
      fogVolumeColor1 = "128 128 128 -2.22768e+038";
      fogVolumeColor2 = "128 128 128 0";
      fogVolumeColor3 = "128 128 128 -1.70699e+038";
      windVelocity = "1 1 0";
      windEffectPrecipitation = "1";
      SkySolidColor = "0.547 0.641 0.789 0";
      useSkyTextures = "1";

      renderBottomTexture = "0";
      noRenderBans = "0";
         locked = "true";

   };

new Sun() {
      azimuth = "0";
      elevation = "35";
      color = "0.988 0.985 0.98 1";
      ambient = "0.5 0.5 0.5 1";
         direction = "0.57735 0.57735 -0.57735";
         position = "0 0 0";
         locked = "true";
         rotation = "1 0 0 0";
         scale = "1 1 1";
   };

new TerrainBlock(Terrain) {
      rotation = "1 0 0 0";
      scale = "1 1 1";
      detailTexture = "~/data/terrains/grass.jpg";
      terrainFile = "./MuseumMission.ter";
      squareSize = "8";
      emptySquares = "99744 443522 443778 444034";
      bumpScale = "1";
      bumpOffset = "0.01";
      zeroBumpScale = "8";
      tile = "1";
         position = "-1024 -1024 0";
         locked = "true";
  };

new WaterBlock(lake) {
      position = "0 -384 -19";
```

```
    rotation = "1 0 0 0";
    scale = "64 32 20";
    UseDepthMask = "0";
    surfaceTexture = "~/data/water/waterblack.jpg";
    ShoreTexture = "~/data/water/waterblack.jpg";
    envMapOverTexture = "~/data/water/waterblack.jpg";
    envMapUnderTexture = "~/data/water/waterblack.jpg";
    liquidType = "Water";
    density = "0.5";
    viscosity = "15";
    waveMagnitude = "1";
    surfaceOpacity = "0.5";
    envMapIntensity = "0.4";
    TessSurface = "50";
    TessShore = "60";
    SurfaceParallax = "0.5";
    FlowAngle = "0";
    FlowRate = "0";
    DistortGridScale = "0.1";
    DistortMag = "0.05";
    DistortTime = "0.5";
    ShoreDepth = "1";
    DepthGradient = "1";
    MinAlpha = "0.9";
    MaxAlpha = "1";
    tile = "1";
    removeWetEdges = "0";
    specularColor = "1 1 1 1";
    specularPower = "6";
    params2 = "0.39 0.39 0.2 0.133";
    seedPoints = "0 0 1 0 1 1 0 1";
    floodFill = "1";
    envMapTexture = "~/data/skies/sunset_0007";
    params0 = "0.32 -0.67 0.066 0.5";
    params1 = "0.63 -2.41 0.33 0.21";
    textureSize = "32 32";
    params3 = "1.21 -0.61 0.13 -0.33";
    Extent = "100 100 10";
 };
//-------------------------------------------------------------
```

4.2 Σχεδιασμός, texturing και εισαγωγή κτιρίων στον εικονικό κόσμο

Το περιβάλλον του εικονικού κόσμου της εφαρμογής περιέχει δύο κτίρια: το γραφείο πληροφοριών και το μουσείο. Εκτός από τα κτίρια αυτά υπάρχει και ένα πλήθος άλλων δομών όπως η γέφυρα, οι σκάλες, οι προστατευτικές μπάρες και οι σκαλωσιές που σχεδιάστηκαν και εισήχθησαν στον εικονικό κόσμο ως interiors, όπως και τα κτίρια. Ο σχεδιασμός και η επιλογή texture για τα κτίρια και τις υπόλοιπες παρεμφερείς δομές έγινε με την βοήθεια του προγράμματος QuArK.

Το QuArK παρέχει μία πληθώρα εργαλείων σχεδιασμού και πολλαπλά παράθυρα για την ταυτόχρονη επισκόπηση διαφορετικών όψεων της σχεδιαζόμενης δομής. Επίσης περιέχει το εργαλείο material browser το οποίο επιτρέπει την επιλογή texture για κάθε έδρα της δομής που σχεδιάζεται. Επιπλέον το πρόγραμμα QuArK

επιτρέπει την προσθήκη αντικειμένων τύπου light entity που αντιστοιχούν σε φωτεινές πήγές με δυνατότητα ρύθμισης της έντασης τους. Οι δομές που σχεδιάζονται στο πρόγραμμα QuArK αποθηκεύονται σε αρχεία τύπου map τα οποία στην συνέχεια εξάγονται με την βοήθεια του exporter map2dif και μετατρέπονται σε αρχεία τύπου dif τα οποία είναι αυτά που τελικά αναγνωρίζει και χρησιμοποιεί το Torque.

Ο κώδικας του αρχείου Museum\data\missions\MuseumMission.κατασκευάζει για κάθε κτίριο ή παρεμφερή δομή ένα αντικείμενο τύπου InteriorInstance στο οποίο αντιστοιχεί προαιρετικά ένα όνομα. Καθένα από τα αντικείμενα αυτά καθορίζει τη θέση, την κλίση και το μέγεθος κάθε δομής, μέσω των παραμέτρων position, rotation και scale, αντίστοιχα.

Επίσης η τιμή του πεδίου interiorFile καθορίζει το αρχείο τύπου dif που περιλαμβάνει τις απαραίτητες πληροφορίες σχετικά με το σχέδιο και τα texture της δομής. Οι παράμετροι useGLLighting και showTerrainInside παρέχουν πληροφορίες για τον φωτισμό του αντικειμένου και για τον αν θα είναι ορατό το terrain στο εσωτερικό του. Οι δομές αυτές βρίσκονται συγκεντρωμένες στον φάκελο με όνομα building που αποτελεί στιγμιότυπο της κλάσης SimGroup και παρουσιάζεται παρακάτω.

Εικόνα 4.5 Το κτίριο του μουσείου μοντέρνας τέχνης

Εικόνα 4.6 Το κτίριο του γραφείου πληροφοριών

Ακολουθούν αποσπάσματα του κώδικά του αρχείου Museum\data\missions\ MuseumMission που αποτελούν παραδείγματα κατασκευής κτιριακών δομών του εικονικό κόσμου.

```
//--------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\data\missions\MuseumMission
//--------------------------------------------------------------------


   new SimGroup(building) {

      new InteriorInstance(bars1) {
         position = "141.5 -418.8 15.5908";
         rotation = "1 0 0 0";
         scale = "1 1 1";
         interiorFile = "~/data/interiors/bars1.dif";
         useGLLighting = "0";
         showTerrainInside = "0";
      };
      new InteriorInstance(OutsideStairs) {
         position = "83 -425.8 14";
         rotation = "1 0 0 0";
         scale = "1.2 1 1";
         interiorFile = "~/data/interiors/escalera.dif";
         useGLLighting = "0";
         showTerrainInside = "0";
      };
```

```
      new InteriorInstance(platform) {
         position = "32.1658 -362.8 0.05";
         rotation = "1 0 0 0";
         scale = "0.5 1.25 0.5";
         interiorFile = "~/data/interiors/platform.dif";
         useGLLighting = "0";
         showTerrainInside = "0";
      };
      new InteriorInstance(bars) {
         position = "131.45 -418.8 15.5908";
         rotation = "1 0 0 0";
         scale = "1 1 1";
         interiorFile = "~/data/interiors/bars.dif";
         useGLLighting = "0";
         showTerrainInside = "0";
      };
      new InteriorInstance(Museum) {
         position = "60 -400 12";
         rotation = "1 0 0 0";
         scale = "1 1 1";
         interiorFile = "~/data/interiors/museo1.dif";
         useGLLighting = "0";
         showTerrainInside = "0";
      };
      new InteriorInstance(house) {
         position = "45 -324.454 2";
         rotation = "0 0 1 180";
         scale = "1 1 1";
         interiorFile = "~/data/interiors/chouse.dif";
         useGLLighting = "0";
         showTerrainInside = "0";
      };
   };
//--------------------------------------------------------
```

4.3 Σχεδιασμός, texturing και εισαγωγή αντικειμένων στον εικονικό κόσμο

Το περιβάλλον του εικονικού κόσμου εκτός από το εξωτερικό περιβάλλον και τα κτίρια περιέχει και μία πληθώρα άλλων αντικειμένων. Τέτοια αντικείμενα είναι τα έπιπλα του γραφείου πληροφοριών, τα εκθέματα του μουσείου, οι προστατευτικές μπάρες του δεύτερου ορόφου, οι βάσεις των γλυπτών, οι προστατευτικές μπάρες των εκθεμάτων και πολλά άλλα. Τα αντικείμενα του εικονικού κόσμου μπορούν να ανήκουν σε μία από της παρακάτω κατηγορίες.

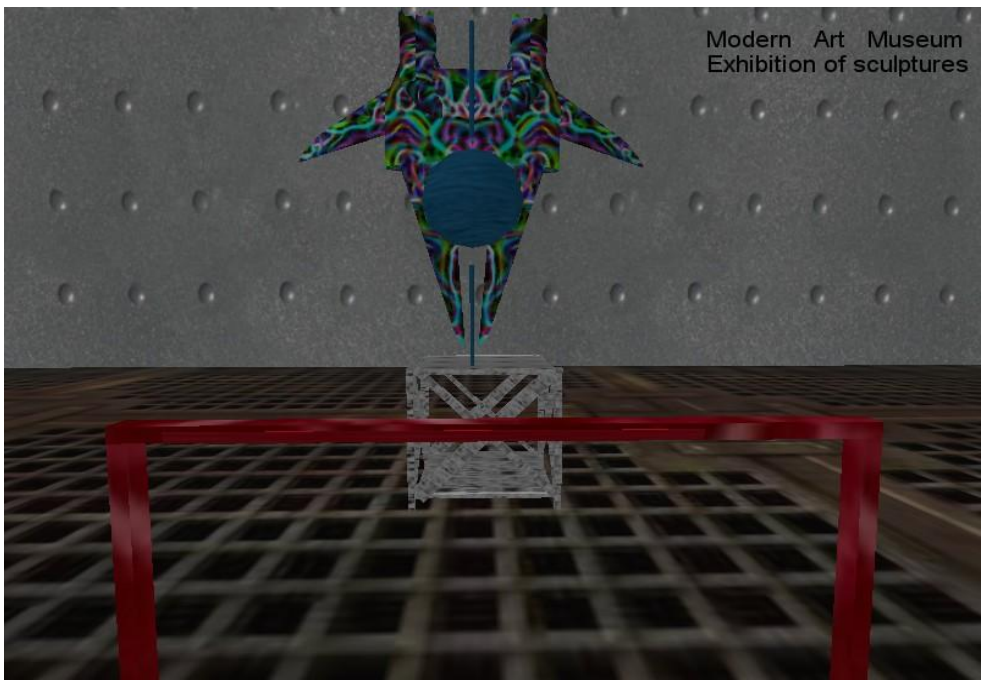
Στην πρώτη κατηγορία ανήκουν τα αντικείμενα, που κωδικοποιούνται από αρχεία τύπου dif. Τα αντικείμενα αυτά εισάγονται στην εφαρμογή με τον τρόπο που περιγράφηκε παραπάνω και αφορά στις κτιριακές δομές. Παραδείγματα αντικειμένων αυτής της κατηγορίας αποτελούν οι βάσεις των γλυπτών, τα έπιπλα του γραφείου πληροφοριών και οι προστατευτικές μπάρες του δεύτερου ορόφου.

Η δεύτερη κατηγορία περιλαμβάνει αντικείμενα των οποίων οι πληροφορίες περιέχονται σε αρχεία τύπου dts. Ο σχεδιασμός και το texturing των αντικειμένων αυτών έγινε με την χρήση του προγράμματος Milkshape. Το Milkshape παρέχει τα απαραίτητα εργαλεία για σχεδιασμό αντικειμένων και πολλαπλά παράθυρα για την ταυτόχρονη επισκόπηση διαφορετικών όψεων του σχεδιαζόμενου αντικειμένου. Κάθε

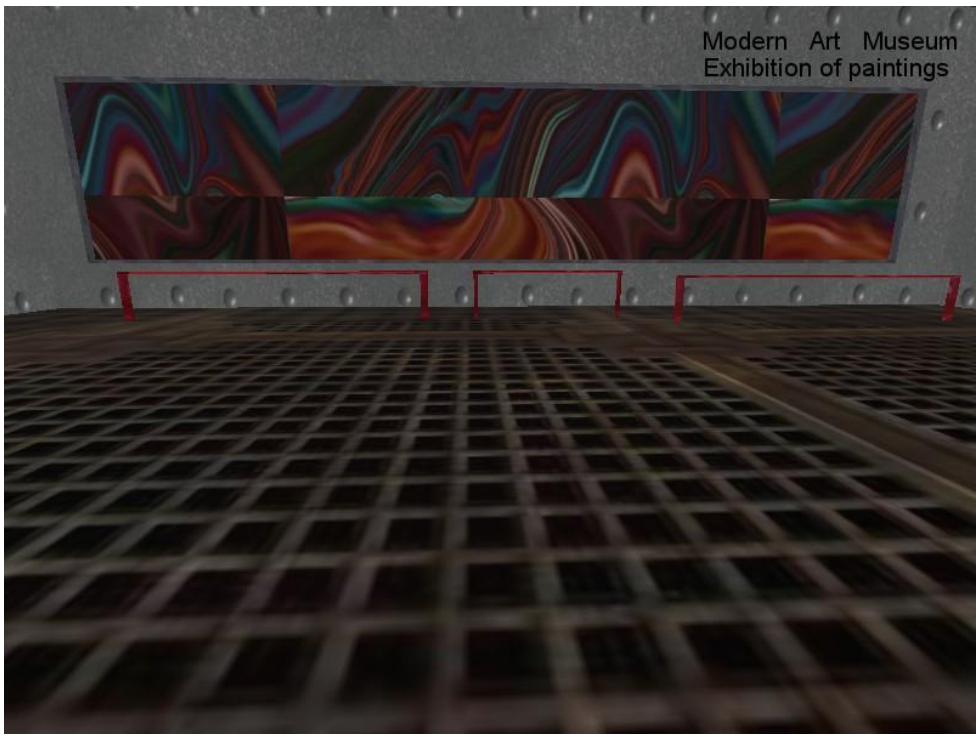
αντικείμενο στο Milkshape αποτελείται από ένα σύνολο σημείων και τριγωνικών επιφανειών. Επίσης παρέχεται η δυνατότητα προσθήκης πλεγμάτων και textures. Με την βοήθεια αυτού του προγράμματος παράγονται αρχεία τύπου ms3d, τα οποία στην συνέχεια μετατρέπονται σε αρχεία τύπου dts με χρήση του exporter ms2dts. Τα αρχεία τύπου dts αναγνωρίζονται από το Torque και μπορούν να εισαχθούν στον εικονικό κόσμο της εφαρμογής. Παραδείγματα τέτοιων αντικειμένων αποτελούν τα εκθέματα του μουσείου.

Για κάθε αντικείμενο τύπου dts προστίθεται στο αρχείο Museum\data\missions\MuseumMission ένα καινούριο αντικείμένο, στιγμιότυπο της κλάσης TSStatic ή της κλάσης StaticShape. Τα αντικείμενα της κλάσης TSStatic καθορίζουν μέσω της παραμέτρου τους shapeName το dts αρχείο που περιλαμβάνει τις πληροφορίες σχετικά με την μορφή του αντικειμένου. Τα αντικείμενα της κλάσης StaticShape καθορίζουν μέσω της παραμέτρου τους dataBlock ένα datablock το οποίο με την σειρά του ορίζει τη μορφή καθώς και την συμπεριφορά του αντικειμένου όταν πρόκειται για αντικείμενο που αλληλεπιδρά με το περιβάλλον. Τα datablock που χρησιμοποιούνται καθώς και οι μέθοδοι που καθορίζουν την συμπεριφορά των αντίστοιχων αντικειμένων βρίσκονται στο αρχείο Museum\server\Items και περιγράφονται αναλυτικά στο κεφάλαιο που αφορά στην αλληλεπίδραση του επισκέπτη με το περιβάλλον του εικονικού κόσμου. Τα στιγμιότυπα της κλάσης StaticShape όπως και αυτά της TSStatic καθορίζουν επίσης την θέση του αντικειμένου στον εικονικό κόσμο την κλίση του ως προς κάθε άξονα και το σχετικό μέγεθός του μέσω των τιμών των παραμέτρων position, rotation και scale, αντίστοιχα.

Για κάθε αντικείμενο τύπου dif προστίθεται στο αρχείο Museum\data\missions\MuseumMission ένα αντικείμένο της κλάσης InteriorInstance, όπως περιγράφηκε στην περίπτωση των κτιρίων.



Εικόνα 4.7 Το γλυπτό 16



Εικόνα 4.8 Ο πίνακας ζωγραφικής 26

Ακολουθούν αποσπάσματα του κώδικά του αρχείου Museum\data\missions\ MuseumMission που αποτελούν παραδείγματα κατασκευής αντικειμένων του εικονικό κόσμου.

```
//--------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\data\missions\MuseumMission
//--------------------------------------------------------------------


new SimGroup(InfoOfficeObjects) {

      new InteriorInstance(Seat) {
         position = "33.5 -80 0.2";
         rotation = "0 0 -1 90";
         scale = "14 14 1";
         interiorFile = "~/data/interiors/pillarseat.dif";
         useGLLighting = "0";
         showTerrainInside = "0";
      };

      new StaticShape(officeBar) {
         position = "42 -313.5 4.065";
         rotation = "1 0 0 0";
         scale = "2.3 0.1 0.05";
         dataBlock = "InfoDeskItem";
      };
      new InteriorInstance(Desk) {
         position = "178 -311.5 1.95";
```

```
         rotation = "1 0 0 0";
         scale = "8 8 0.72";
         interiorFile = "~/data/interiors/pillardsk.dif";
         useGLLighting = "0";
         showTerrainInside = "0";
      };
   };


new SimGroup(sculptures) {

     new TSStatic(sculpture6) {
         position = "78.0764 -412.2 3";
         rotation = "1 0 0 0";
         scale = "1.5 1.5 1.5";
         shapeName = "~/data/shapes/scc.dts";
      };
      new TSStatic(sculpture4) {
         position = "63.5 -436.4 3.2";
         rotation = "-1 0 0 90";
         scale = "2.5 2.5 2.5";
         shapeName = "~/data/shapes/s4.dts";
      };
};

new SimGroup(pictureBars) {

      new StaticShape(wbar22) {
         position = "84.5 -411.2 13.4";
         rotation = "1 0 0 0";
         scale = "0.1 12 0.1";
         dataBlock = "Base22Item";
      };
      new StaticShape(wbar23) {
         position = "84.5 -429 13.4";
         rotation = "1 0 0 0";
         scale = "0.1 12 0.1";
         dataBlock = "Base23Item";
      };

      new StaticShape(wbar21) {
         position = "69.1 -399.86 13.56";
         rotation = "1 0 0 0";
         scale = "12 0.1 0.1";
         dataBlock = "Base21Item";
      };
      new StaticShape(wbar21a) {
         position = "78.1 -399.86 13.56";
         rotation = "1 0 0 0";
         scale = "12 0.1 0.1";
         dataBlock = "Base21Item";

      };
     new InteriorInstance(pbar1a) {
         position = "213 -415 14.3";
         rotation = "1 0 0 0";
         scale = "3 1 1";
         interiorFile = "~/data/interiors/objectbar.dif";
         useGLLighting = "0";
         showTerrainInside = "0";
```

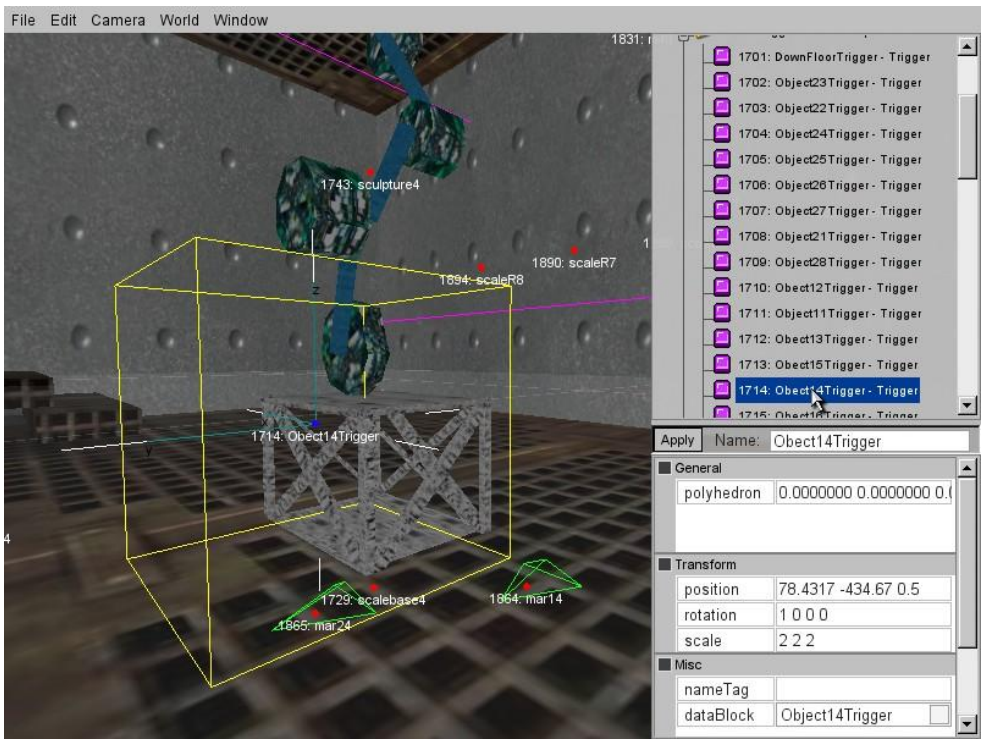
```
        };

    };
//---------------------------------------------------------------------------------------------
```

4.4 Κατασκευή αντικειμένων ελέγχου

Πέραν των αντικειμένων, των κτιρίων και των στοιχείων του περιβάλλοντος που διαμορφώνουν το ορατό αποτέλεσμα του εικονικού κόσμου, εξίσου σημαντικά είναι και τα αντικείμενα ελέγχου τα οποία συνήθως δεν είναι ορατά από τον χρήστη κατά την διάρκεια εκτέλεσης της εφαρμογής. Η ύπαρξη των αντικειμένων αυτών έχει ως σκοπό την αποθήκευση πληροφοριών σε συγκεκριμένα σημεία του περιβάλλοντος για τον έλεγχο δραστηριοτήτων και τον καθορισμό της ροής εκτέλεσης της εφαρμογής. Τέτοια αντικείμενα είναι τα triggers, τα αντικείμενα της κλάσης SpawnSphere, τα paths και οι markers.

4.4.1 Area triggers

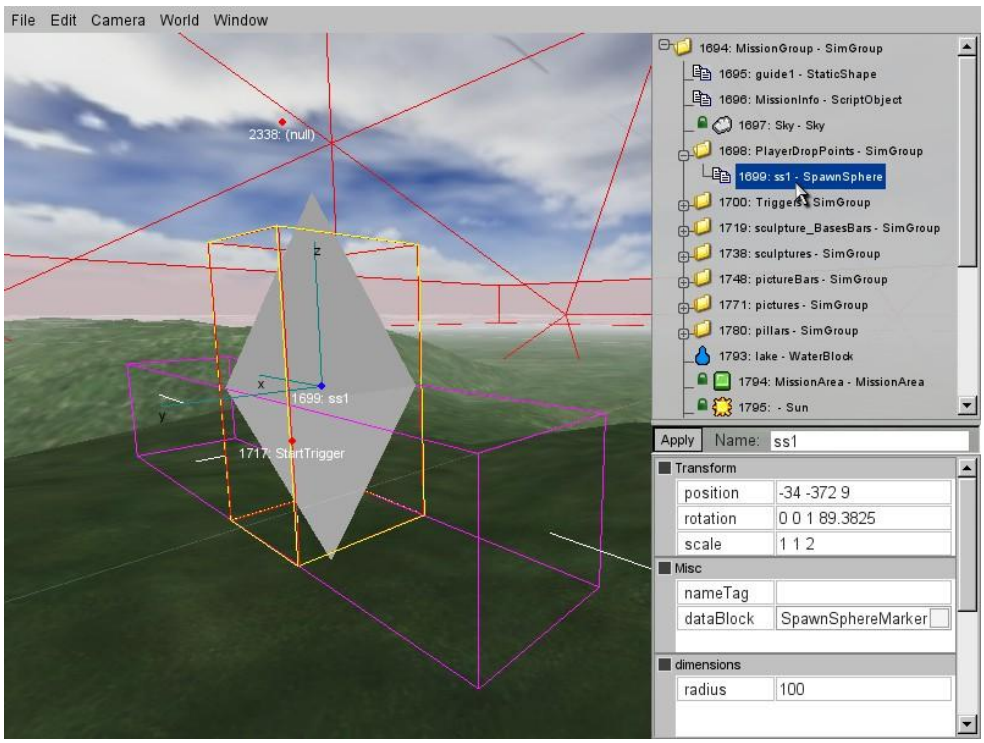
Τα χωρικά triggers που χρησιμοποιούνται κατά την διάρκεια της εφαρμογής οριοθετούν μία περιοχή του εικονικού κόσμου στην οποία προσδίδονται στην συνέχεια συγκεκριμένες ιδιότητες και καθορισμένη συμπεριφορά κατά την αλληλεπίδραση της με τους χαρακτήρες της εφαρμογής. Η διαδραστική συμπεριφορά των triggers καθορίζεται από τα datablock που χρησιμοποιούν. Οι μέθοδοι onEnterTrigger, onLeaveTrigger και onTickTrigger καθορίζουν τις ενέργειες που εκτελούνται κάθε φορά που κάποιος χαρακτήρας ή αντικείμενο εισέρχεται στον χώρο του trigger, εξέρχεται ή παραμένει σε αυτόν, αντίστοιχα. Συγκεκριμένα για κάθε trigger κατασκευάζεται ένα νέο αντικείμενο που αποτελεί στιγμιότυπο της κλάσης Trigger και περιλαμβάνει πληροφορίες σχετικά με το σχήμα του, την θέση του αντικειμένου στον εικονικό κόσμο την κλίση του ως προς κάθε άξονα και το σχετικό μέγεθός του, μέσω των τιμών των παραμέτρων polyhedron, position, rotation και scale, αντίστοιχα. Επίσης αντιστοιχίζεται στο αντικείμενο αυτό το datablock που θα καθορίσει την συμπεριφορά του μέσω της τιμής της μεταβλητής dataBlock. Τα triggers που χρησιμοποιήθηκαν κατά την διάρκεια της εφαρμογής περιγράφονται αναλυτικότερα στο κεφάλαιο που αφορά στην αλληλεπίδραση των χαρακτήρων της εφαρμογής με το περιβάλλον αυτής.



Εικόνα 4.9 Κατασκευή trigger στον world editor

4.4.2 SpawnSphere

Τα αντικείμενο της κλάσης SpawnSphere καθορίζουν σημεία του εικονικού κόσμου που μπορούν να χρησιμεύσουν ως σημεία εμφάνισης των χαρακτήρων της εφαρμογής. Στην συγκεκριμένη εφαρμογή κατασκευάζεται μόνο ένα στιγμιότυπό της κλάσης SpawnSphere που ορίζει της συντεταγμένες του σημείου όπου εμφανίζεται ο επισκέπτης κατά την εκκίνηση της πλοήγησής του στον εικονικό κόσμο.



Εικόνα 4.10 Κατασκευή του αντικειμένου SpawnSphere στον world editor

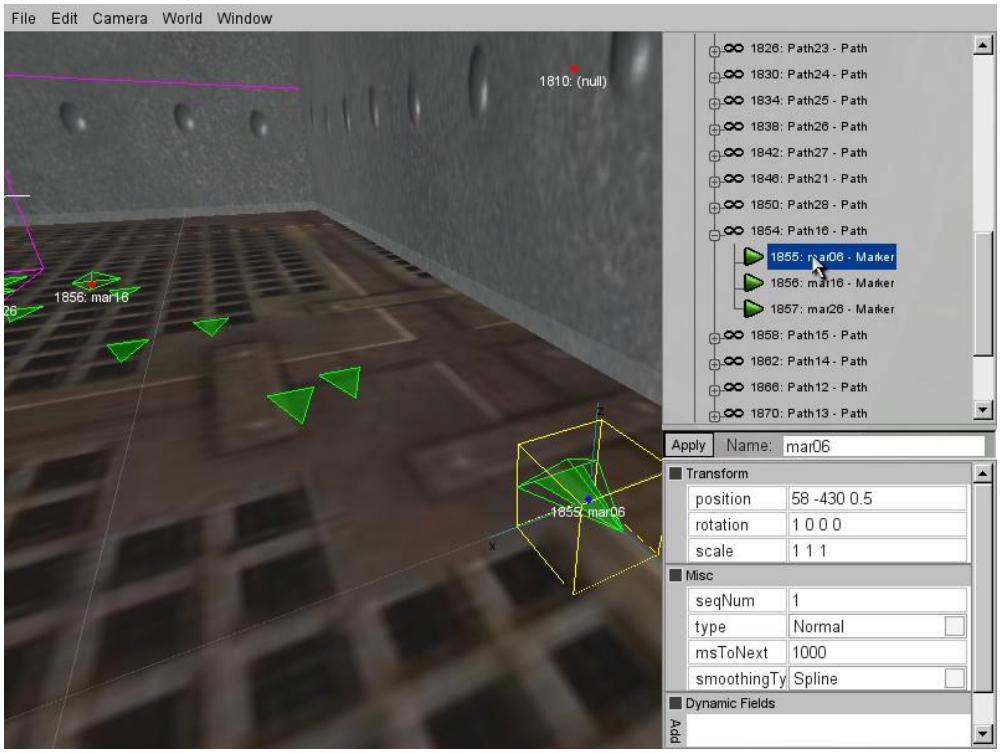
4.4.3 Paths και markers

Τα paths είναι μονοπάτια που αποτελούνται από κόμβους που ονομάζονται markers. Με την κατασκευή αντικειμένων τύπου path καθορίζεται μία διαδρομή μέσα στον εικονικό κόσμο που μπορεί στην συνέχεια να ακολουθήσει κάποιος χαρακτήρας που ανήκει στην κλάση AIPlayer. Η κλάση AIPlayer έχει μια σειρά μεθόδων που ορίζουν την κίνηση ενός χαρακτήρα AIPlayer πάνω σε ένα μονοπάτι. Οι βασικότερες από αυτές τις μεθόδους είναι οι spawnOnPath, η followPath, η moveToNextNode και η moveToNode των οποίων ο κώδικας βρίσκεται στο αρχείο Museum\server\aiPlayer. Συγκεκριμένα η μέθοδος spawnOnPath εμφανίζει έναν χαρακτήρα στο σημείου όπου βρίσκεται ο πρώτος κόμβος του μονοπατιού που δέχεται σαν όρισμα. Η μέθοδος followPath αναγκάζει τον χαρακτήρα να κινηθεί κατά μήκος του μονοπατιού που ορίζεται από την αντίστοιχη παράμετρο και αφού διατρέξει όλους τους κόμβους του διαδοχικά, να καταλήξει στο κόμβο που ορίζεται ως τελικός. Η μέθοδος moveToNextNode αναγκάζει τον χαρακτήρα να κινηθεί στον επόμενο κατά σειρά κόμβο του μονοπατιού στο οποίο βρίσκεται και η μέθοδος moveToNode τον αναγκάζει να κινηθεί στο κόμβο που ορίζεται από την αντίστοιχη παράμετρο της μεθόδου.

Για κάθε μονοπάτι προστίθεται στο αρχείο Museum\data\missions\MuseumMission ένα νέο αντικείμενο της κλάσης Path. Το αντικείμενο αυτό περιλαμβάνει την παράμετρο isLooping που καθορίζει αν το μονοπάτι θα είναι κυκλικό ή όχι και ένα ή περισσότερα αντικείμενα της κλάσης Marker που αποτελούν τους κόμβους του μονοπατιού. Τα στιγμιότυπα της κλάσης Marker εκτός από πληροφορίες σχετικά με την θέση τους και την μορφή τους περιέχουν και έναν αύξοντα αριθμό που καθορίζει την σειρά τους στο μονοπάτι. Ο

αριθμός αυτός αποτελεί την τιμή της παραμέτρου seqNum των αντικειμένων τύπου Marker και παίζει σημαντικό ρόλο κατά την κίνηση του χαρακτήρα πάνω στο μονοπάτι.

Στον εικονικό κόσμο του μουσείου κατασκευάστηκάν ισάριθμά με τα εκθέματα του μουσείου, μονοπάτια. Σκοπός τους είναι να οδηγούν τον ξεναγό του μουσείου μπροστά από κάθε έκθεμα όταν ο επισκέπτης ζητήσει περαιτέρω πληροφορίες για κάποιο από τα εκθέματα αυτά.



Εικόνα 4.11 Οι markers του path 16.

Ακολουθεί ο κώδικάς που υλοποιεί κάποια από τα triggers της εφαρμογής, το αντικείμενο της κλάσης SpawnSphere και κάποια από τα paths με τους αντίστοιχους markers.

```
//--------------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\data\missions\MuseumMission
//--------------------------------------------------------------------------


new SimGroup(Triggers) {

new Trigger(Object11Trigger) {
         position = "62 -411 0.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object11Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
```

```
0.0000000 1.0000000";
      };
      new Trigger(infoOfficeTrigger) {
         position = "32.5 -309 2";
         rotation = "1 0 0 0";
         scale = "16 16 6";
         dataBlock = "InfoHouseTrigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(StartTrigger) {
         position = "-34.5 -370 8";
         rotation = "0 0 -1 0.0395647";
         scale = "1 5 1";
         dataBlock = "StartTrigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(DownFloorTrigger) {
         position = "54.2 -402 -1.93715e-007";
         rotation = "1 0 0 0";
         scale = "33 43 5";
         dataBlock = "ExpoOneTrigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(UpFloorTrigger) {
         position = "55 -396 9.6";
         rotation = "1 0 0 0";
         scale = "33 55 11";
         dataBlock = "ExpoTwoTrigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
   };


new SimGroup(PlayerDropPoints) {

      new SpawnSphere(ss1) {
         position = "-34 -372 9";
         rotation = "0 0 1 89.3825";
         scale = "1 1 2";
         dataBlock = "SpawnSphereMarker";
         radius = "100";
         sphereWeight = "100";
         indoorWeight = "100";
         outdoorWeight = "100";
            locked = "False";
            lockCount = "0";
            homingCount = "0";
      };
   };


new SimGroup(Paths) {
```

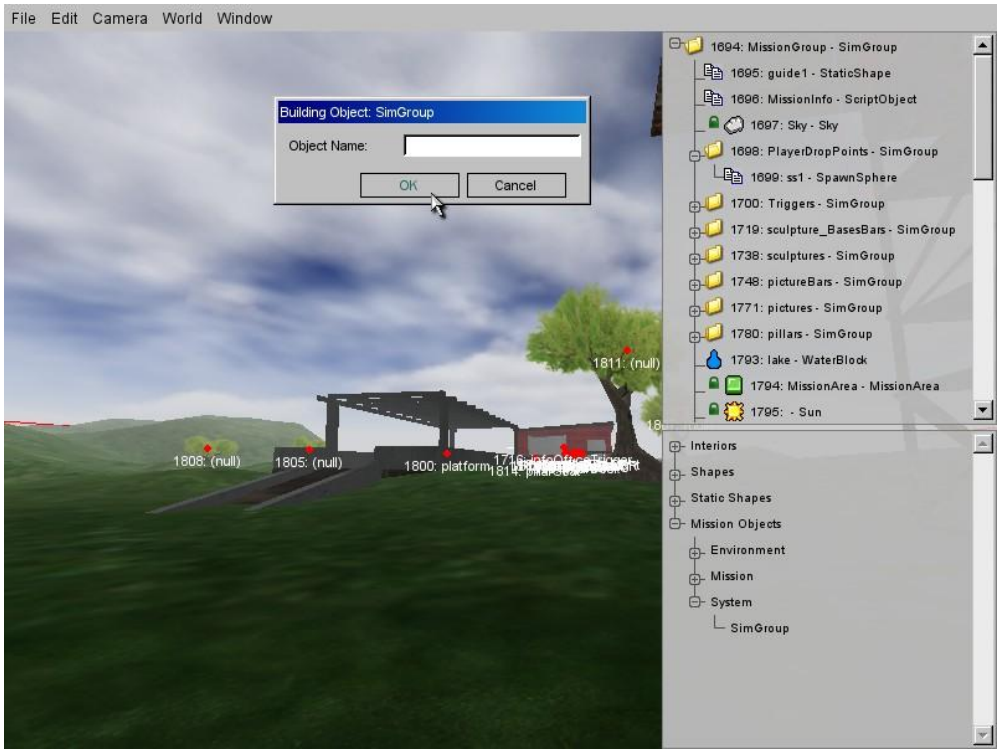
```
      new Path(Path11) {
         isLooping = "1";

         new Marker(mar01) {
            position = "60 -419 0.3";
            rotation = "1 0 0 0";
            scale = "1 1 1";
            seqNum = "1";
            type = "Normal";
            msToNext = "1000";
            smoothingType = "Spline";
         };
         new Marker(mar11) {
            position = "62.3 -412 1";
            rotation = "1 0 0 0";
            scale = "1 1 1";
            seqNum = "2";
            type = "Normal";
            msToNext = "1000";
            smoothingType = "Spline";
         };
         new Marker(mar21) {
            position = "63.5 -412.5 1";
            rotation = "1 0 0 0";
            scale = "1 1 1";
            seqNum = "3";
            type = "Normal";
            msToNext = "1000";
            smoothingType = "Spline";
         };
      };
   };

//--------------------------------------------------------------------
```

4.5 Κατασκευή αντικειμένων που συμβάλλουν στην οργάνωση της εφαρμογής

Τα αντικείμενα αυτής της κατηγορίας δεν τοποθετούνται στο περιβάλλον της εφαρμογής και έχουν σκοπό την οργάνωση των υπόλοιπων αντικειμένων του περιβάλλοντος και την αποθήκευση πληροφοριών σχετικά με το είδος της εφαρμογής. Τέτοια αντικείμενα είναι τα αντικείμενα των κλάσεων SimGroupκαι ScriptObject.

Το αντικείμενο της κλάσης ScriptObject με όνομα MissionInfo, περιλαμβάνει πληροφορίες σχετικά με το είδος της εφαρμογής, οι οποίες εμφανίζονται κατά το στάδιο φόρτωσής. Οι πληροφορίες περιλαμβάνουν το όνομα της εφαρμογής και ένα μήνυμα κειμένου που καθορίζει ο σχεδιαστής του εικονικού κόσμου.

Τα στιγμιότυπά της κλάσης SimGroup λειτουργούν σαν φάκελοι που βοηθούν στην ομαδοποίηση παρεμφερών αντικειμένων και κατ' επέκταση στην αναγνωσιμότητα του αρχείου Museum\data\missions\MuseumMission που περιέχει τα συστατικά στοιχεία του περιβάλλοντος της εφαρμογής. Επίσης διευκολύνουν τον προγραμματισμό της εφαρμογής, αφού με την βοήθεια τους μπορούν να ομαδοποιηθούν τα αντικείμενα του εικονικού κόσμου και στη συνέχεια τα περιεχόμενα της κάθε ομάδας μπορούν να αντιμετωπιστούν με παρόμοιο τρόπο.



Εικόνα 4.12 Κατασκευή αντικειμένου SimGroup στον world editor.

Ο κώδικάς που υλοποιεί το αντικείμενο της κλάσης ScriptObject της εφαρμογής φαίνεται παρακάτω. Παραδείγματα χρήσης αντικειμένων της κλάσης SimGroup έχουν δοθεί παραπάνω, όπως στην περίπτωση παρουσίασης του κώδικα που υλοποιεί τις κτιριακές δομές της εφαρμογής.

```
//--------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\data\missions\MuseumMission
//--------------------------------------------------------------------


   new ScriptObject(MissionInfo) {
         desc0 = "Created by Antonopoulou C.";
         name = "Modern Art Museum";
   };
```

5. ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΚΑΙ ΚΙΝΗΣΗ ΧΑΡΑΚΤΗΡΩΝ, ΕΙΣΑΓΩΓΗ ΤΩΝ ΧΑΡΑΚΤΗΡΩΝ ΣΤΟΝ ΕΙΚΟΝΙΚΟ ΚΟΣΜΟ.

Η μοντελοποίηση το texturing και το animation των χαρακτήρων έγινε με την βοήθεια των προγραμμάτων Milkshape, UVMapper, Photoshop, CharacterFX και TorqueShowToolPro.

Τα μοντέλα των τριών χαρακτήρων κατασκευάστηκαν ως μοντέλα συνεχούς πλέγματος (continuous-mesh models), δηλαδή το σώμα του χαρακτήρα αποτελείται από μία ενιαία επιφάνεια σε αντίθεση με τα μοντέλα segmented-mesh τα οποία αποτελούνται από πολλά διακριτά αντικείμενα ή πλέγματα που αντιστοιχούν στα διάφορα μέλη του σώματος του μοντέλου. Η μοντελοποίηση πραγματοποιήθηκε στο πρόγραμμα Milkshape. Τα μοντέλα που κατασκευάζονται στο πρόγραμμα αυτό αποτελούνται από ένα σύνολο σημείων-κορυφών και έναν μεγάλο αριθμό τριγωνικών επιφανειών. Το πρόγραμμα περιλαμβάνει ένα πλήθος εργαλείων για την επιλογή, κίνηση, περιστροφή, συνένωση και εκτέλεση πολλών άλλων ενεργειών πάνω στα σημεία και τις επιφάνειες με σκοπό την διαμόρφωση της τελικής μορφής του μοντέλου. Επίσης υπάρχει η δυνατότητα ταυτόχρονης εμφάνισης διαφορετικών όψεων του μοντέλου που διευκολύνει την διαδικασία της μοντελοποίησης.

Όταν ολοκληρωθεί η μοντελοποίηση του χαρακτήρα αρχίζει η διαδικασία του texturing. Για τον σκοπό αυτό το μοντέλο του χαρακτήρα εξάγεται από το Milkshape στο UVMapper μέσω της επιλογής Wavefront Obj. Στο πρόγραμμα UVMapper η επιφάνεια που καλύπτει το κάθε μέλος του σώματος του μοντέλου ξετυλίγεται και το ανάπτυγμα που προκύπτει αποθηκεύεται σε μία εικόνα τύπου jpeg ή bmp. Η εικόνα αυτή μπορεί στη συνέχεια να επεξεργαστεί με την βοήθεια του προγράμματος Photoshop. Με το πρόγραμμα αυτό μπορούμε να δώσουμε στο πρόσωπο, το δέρμα και τα ρούχα τα χαρακτηριστικά που επιθυμούμε, ώστε το μοντέλο του χαρακτήρα να πάρει την τελική του μορφή.

Μετά την ολοκλήρωση της μοντελοποίησης και του texturing, το μόνο που μένει είναι ο σχεδιασμός της κίνησης του χαρακτήρα. Η διαδικασία αυτή είναι αρκετά περίπλοκη λόγω της πολυπλοκότητας της ανθρώπινης κίνησης. Για την πραγματοποίηση της απαιτείται η εξαγωγή του χαρακτήρα από το Milkshape, ως αρχείο ASCII, και η εισαγωγή του στο CharacterFX.

Το πρώτο στάδιο για την δημιουργία του animation είναι η κατασκευή του σκελετού και η αντιστοίχιση των σημείων του μοντέλου στους κατάλληλους κόμβους του σκελετού. Για την κατασκευή του σκελετού και την σωστή τοποθέτηση των κόμβων πρέπει να ληφθούν υπόψη οι κινήσεις που επιθυμούμε να εκτελεί ο χαρακτήρας. Για την πραγματοποίηση αυτών των κινήσεων είναι απαραίτητος ο ορισμός μίας ιεραρχίας μεταξύ των κόμβων. Έτσι αν κατά τον σχεδιασμό του animation κινηθεί ένας κόμβος που βρίσκεται ψηλά στη ιεραρχία των κόμβων, ακολουθούν την κίνηση του και όλοι οι χαμηλότερης ιεραρχίας κόμβοι που εξαρτώνται από αυτόν. Επίσης όταν κινείται ένας κόμβος ακολουθούν την κίνηση του και τα σημεία του σώματος του μοντέλου που έχουν αντιστοιχηθεί στον συγκεκριμένο κόμβο μέσω της διαδικασίας του assignment που πραγματοποιείται μετά την δημιουργία του σκελετού.

Οι παραπάνω διαδικασίες υλοποιούνται στο object mode του CaracterFX., ενώ ο σχεδιασμός της κίνησης πραγματοποιείται στο animation mode. Το animation mode περιλαμβάνει το παράθυρο του keyframer το οποίο περιέχει όλα τα frames που συνιστούν την κίνηση του χαρακτήρα. Η στάση του σώματος του χαρακτήρα σε κάθε frame της κίνησης μπορεί να οριστεί επιλέγοντας το αντίστοιχο frame και

χρησιμοποιώντας τα εργαλεία κίνησης και περιστροφής των κόμβων του σκελετού που παρέχει το πρόγραμμα. Στην πραγματικότητα όμως με τον παραπάνω τρόπο ορίζονται τα κομβικά frames της κάθε κίνησης ενώ τα ενδιάμεσα frames υπολογίζονται από το ίδιο το πρόγραμμα με χρήση παρεμβολής. Το animation mode παρέχει επιλογές που καθορίζουν το είδος της παρεμβολής, την ταχύτητα με την οποία θα εναλλάσσονται τα frames και άλλες παραμέτρους της κίνησης.

Όταν καθοριστούν όλες οι κινήσεις γίνεται εξαγωγή του χαρακτήρα στο Torque μέσω του Milkshape. Για τον σκοπό αυτό το αρχείο αρχικά εξάγεται ως αρχείο ASCII με επέκταση txt και εισάγεται στο Milkshape. Οι ακολουθίες κινήσεων και τα frames που αποτελούν την κάθε κίνηση ορίζονται στο Milkshape ως ειδικού τύπου materials. Συγκεκριμένα για κάθε ακολουθία κίνησης δημιουργείται ένα material που καθορίζει το όνομα της κίνησης τα frames που την αποτελούν, την ταχύτητα με την οποία θα εναλλάσσονται τα frames κατά την εκτέλεση της κίνησης στον εικονικό κόσμο καθώς και το αν η κίνηση είναι κυκλική ή όχι. Η ταχύτητα καθορίζεται από την τιμή της παραμέτρου fps (frames per second) που ορίζει τον αριθμό των frames της κάθε κίνησης που θα εκτελούνται κάθε δευτερόλεπτο. Αν η κίνηση οριστεί ως κυκλική τότε κάθε φορά που εκτελούνται όλα τα frames της αντίστοιχης ακολουθίας αρχίζει από την αρχή η επανεκτέλεσή τους. Αφού οριστούν όλες οι ακολουθίες κινήσεων, ο χαρακτήρας μπορεί να εξαχθεί μέσω του Torque DTS plus exporter. Ο exporter παράγει ένα εκτελέσιμο αρχείο dts που κωδικοποιεί την μορφή του χαρακτήρα και ισάριθμα με τις κινήσεις του χαρακτήρα εκτελέσιμα αρχεία dsq που κωδικοποιούν τις κινήσεις.

Ο χαρακτήρας και οι ακολουθίες κίνησης του μπορούν να δοκιμαστούν πριν την εισαγωγή τους στον εικονικό κόσμο στο πρόγραμμα TorqueShowToolPro. Το πρόγραμμα αυτό δίνει την δυνατότητα ελέγχου, προκειμένου να διαπιστωθεί αν οι ακολουθίες κινήσεων εκτελούνται κατά τον επιθυμητό τρόπο πριν την εισαγωγή του χαρακτήρα στο περιβάλλον του εικονικού κόσμου.

Για την κατασκευή του αντικειμένου κάθε χαρακτήρα απαιτείται ένα datablock τύπου PlayerData που ορίζει την μορφή του χαρακτήρα καθώς και ένα datablock τύπου TSShapeConstructor που αντιστοιχεί κάθε εκτελέσιμο dsq αρχείο με μία κίνηση που αναγνωρίζει το Torque.

Στο περιβάλλον του εικονικού κόσμου του μουσείου υπάρχουν τρεις κατηγορίες χαρακτήρων. Στην πρώτη κατηγορία ανήκει ο επισκέπτης που αποτελεί στιγμιότυπο της κλάσης Player. Μέσω του binding, δίνεται η δυνατότητα στον χρήστη να κινεί τον χαρακτήρα αυτό προς όποια κατεύθυνση επιθυμεί. Το δεύτερο είδος περιλαμβάνει τον χαρακτήρα του ξεναγού του μουσείου ο οποίος αποτελεί στιγμιότυπο της κλάσης AIPlayer. Ο χαρακτήρας αυτός κινείται με προκαθορισμένο τρόπο όπως ορίζεται από τον κώδικα της εφαρμογής, χωρίς να δίνεται στον χρήστη η δυνατότητα να τον κινήσει άμεσα όπως επιθυμεί. Τέλος, ο υπάλληλος του μουσείου που βρίσκεται στο γραφείο πληροφοριών δεν κινείται και επομένως δεν είναι αναγκαίο να δημιουργηθεί animation για αυτόν τον χαρακτήρα. Ο χαρακτήρας αυτός έχει υλοποιηθεί ως στατική μορφή.

5.1 Χαρακτήρας επισκέπτη.

Ο χαρακτήρας του επισκέπτη είναι ο χαρακτήρας τον οποίο μπορεί να κινήσει ο χρήστης της εφαρμογής και μέσω του οποίου μπορεί να περιηγηθεί στον εικονικό

κόσμο του μουσείου. Η μοντελοποίηση, το texturing και ο σχεδιασμός της κίνησης του χαρακτήρα αυτού ακολούθησε την διαδικασία που περιγράφηκε παραπάνω.

Ο χαρακτήρας του επισκέπτη κατασκευάζεται κατά την έναρξη της πλοήγησης στον εικονικού κόσμου μέσω της συνάρτησης createPlayer του αντικειμένου GameConnection. Εμφανίζεται στο σημείο που προσδιορίζει το αντικείμενο ss1 που αποτελεί στιγμιότυπο της κλάσης SpawnSphere και η θέση του καθορίζεται κατά την κατασκευή του στο αρχείο Museum\data\missions\MuseumMission.

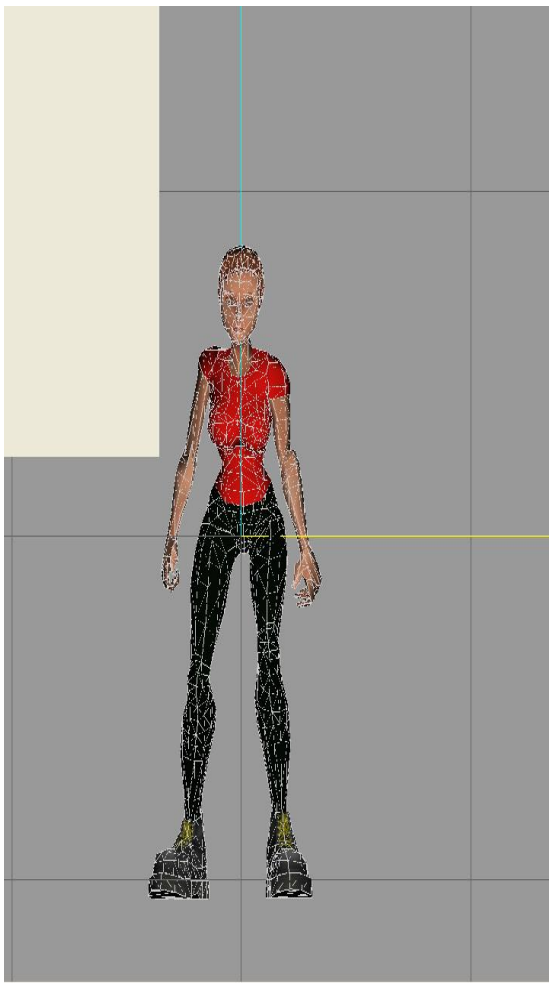
Για τον χαρακτήρα του επισκέπτη έχουν σχεδιαστεί 4 ακολουθίες animation που υλοποιούν την στάση του χαρακτήρα όταν δεν κινείται, το τρέξιμο προς τα μπροστά, την προς τα πίσω κίνηση και την πλάγια κίνηση του χαρακτήρα. Οι ακολουθίες αυτές έχουν αντιστοιχηθεί σε συγκεκριμένα πλήκτρα του υπολογιστή, έτσι ώστε να είναι δυνατή η κίνηση του χαρακτήρα από τον χρήστη με πάτημα των αντίστοιχων πλήκτρων. Συγκεκριμένα έχουν αντιστοιχηθεί τα πλήκτρα w, s, a και d στην προς τα μπρος κίνηση του χαρακτήρα, στην προς τα πίσω κίνηση, στην πλάγια αριστερά και στην πλάγια δεξιά κίνηση αντίστοιχα.

Η μορφή του χαρακτήρα ορίζεται από το datablock τύπου PlayerData με όνομα PlayerShape. Η παράμετρος shapeFile του datablock αυτού παίρνει ως τιμή το αρχείο Museum/data/shapes/visitor/visitor.dts που αποτελεί το εκτελέσιμο αρχείο dts που περιέχει την μορφή του χαρακτήρα.

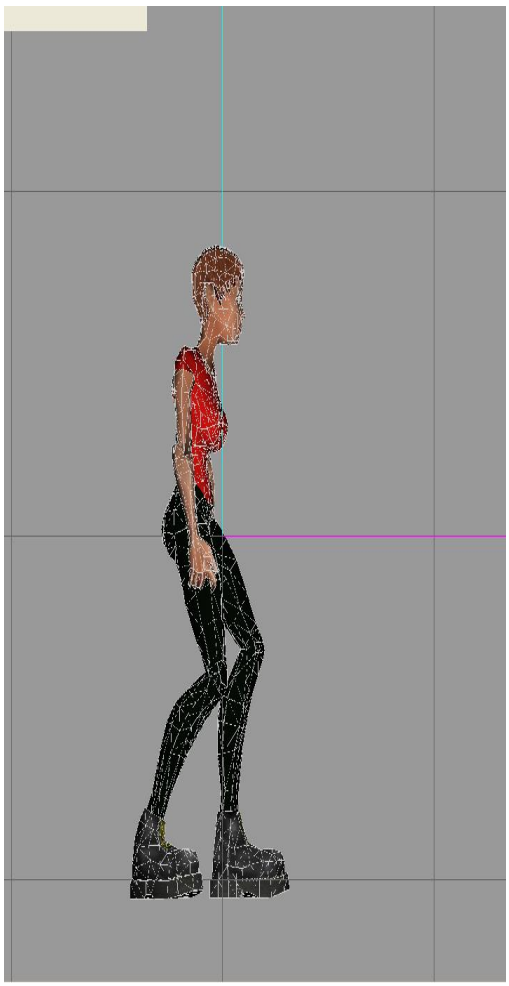
Το datablock PlayerShape υλοποιεί τις μεθόδους onAdd, onRemove και onNewDataBlock. Οι παραπάνω μέθοδοι μπορούν να χρησιμοποιηθούν για την εκτέλεση ορισμένων ενεργειών όταν το συγκεκριμένο datablock διαβάζεται για πρώτη φορά από το Torque, όταν καταστρέφεται ένα αντικείμενο που χρησιμοποιεί το datablock ή όταν δημιουργείται ένα νέο αντικείμενο που χρησιμοποιεί το datablock, αντίστοιχα.

Εκτός από το datablock PlayerShape απαιτείται και η κατασκευή ενός datablock τύπου TSShapeConstructor που θα αντιστοιχεί τα εκτελέσιμα αρχεία dsq που περιγράφουν τις κινήσεις του χαρακτήρα στις αντίστοιχες κινήσεις που αναγνωρίζει το Torque. Το datablock αυτό στην περίπτωση του χαρακτήρα του επισκέπτη έχει το όνομα PlayerDts και αντιστοιχίζει τα αρχεία visitor_root.dsq, visitor_forward.dsq, visitor_back.dsq, visitor_side.dsq του φακέλου Museum\data\shapes\visitor στις κινήσεις root, run, back και side, αντίστοιχα.

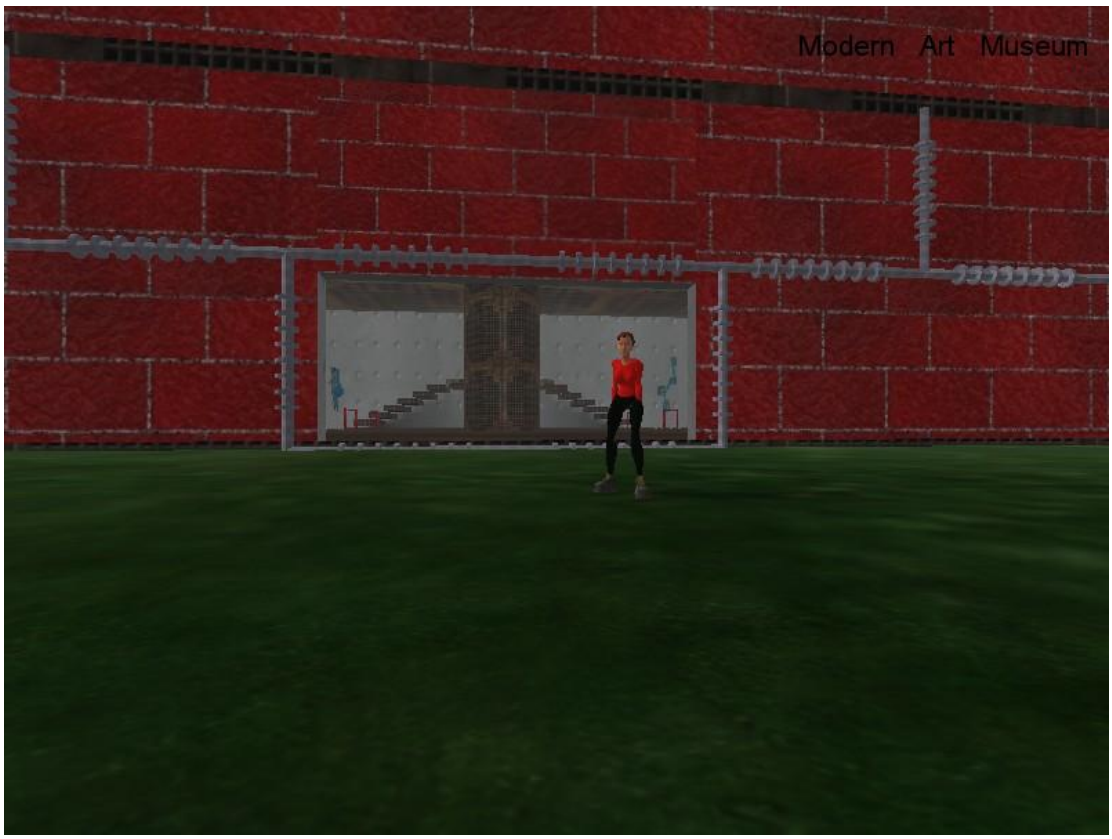
Τα εκτελέσιμα αρχεία dsq που κωδικοποιούν την κίνηση του χαρακτήρα καθώς και το αρχείο dts που κωδικοποιεί την μορφή του βρίσκονται στον φάκελο Museum\data\shapes\visitor. Στον ίδιο φάκελο βρίσκονται αποθηκευμένα και τα textures
που χρησιμοποιεί το αρχείο dts για την διαμόρφωση της τελικής μορφής του χαρακτήρα.



Εικόνα 5.1 Στιγμιότυπο από την μοντελοποίηση του χαρακτήρα του επισκέπτη στο Milkshape.



Εικόνα 5.2 Στιγμιότυπο από την μοντελοποίηση του χαρακτήρα του επισκέπτη στο Milkshape.



Εικόνα 5.3 Ο χαρακτήρας του επισκέπτη στο περιβάλλον του εικονικού κόσμου.

Ο κώδικάς που ορίζει τα datablocks που απαιτούνται για την κατασκευή του χαρακτήρα και την υλοποίηση των αντίστοιχων συναρτήσεων βρίσκεται στο αρχείο Museum\server\visitor και παρατίθεται στην συνέχεια..

```
//-----------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\server\visitor
//-----------------------------------------------------------------------

// Load dts shapes and merge animations

datablock TSShapeConstructor(PlayerDts)
{
   baseShape = "~/data/shapes/visitor/visitor.dts";
   sequence0 = "~/data/shapes/visitor/visitor_root.dsq root";
   sequence1 = "~/data/shapes/visitor/visitor_forward.dsq run";
   sequence2 = "~/data/shapes/visitor/visitor_back.dsq back";
   sequence3 = "~/data/shapes/visitor/visitor_side.dsq side";

};

datablock PlayerData(PlayerShape)
{
   renderFirstPerson = false;
   shapeFile = "~/data/shapes/visitor/visitor.dts";
};
```

```
//--------------------------------------------------------------------
// PlayerShape Datablock methods
//--------------------------------------------------------------------

function PlayerShape::onAdd(%this,%obj)
{
   // Called when the PlayerData datablock is first 'read' by the
engine (executable)
}

function PlayerShape::onRemove(%this, %obj)
{
   if (%obj.client.player == %obj)
      %obj.client.player = 0;
}

function PlayerShape::onNewDataBlock(%this,%obj)
{
   // Called when this PlayerData datablock is assigned to an object
}

//--------------------------------------------------------------------
```

5.2 Ο χαρακτήρας του ξεναγού

Ο χαρακτήρας του ξεναγού εμφανίζεται κατά την διάρκεια εκτέλεσης της εφαρμογής όταν το επιλέξει ο χρήστης, προκειμένου να έχει λεπτομερέστερη πληροφόρηση σχετικά με κάποιο από τα εκθέματα του μουσείου. Για την μοντελοποίηση το texturing και τον σχεδιασμό της κίνησης του χαρακτήρα ακολουθήθηκε η διαδικασία που περιγράφηκε στην αρχή του κεφαλαίου όπως και στην περίπτωση του μοντέλου του επισκέπτη.

Για την κίνηση του ξεναγού σχεδιάστηκαν οι ακολουθίες που κωδικοποιούν την στάση του χαρακτήρα όταν δεν κινείται, το τρέξιμο προς τα μπροστά, την προς τα πίσω κίνηση και την πλάγια κίνηση του χαρακτήρα. Δεν υπάρχει αντιστοίχηση των κινήσεων αυτών σε πλήκτρα, αφού ο χαρακτήρας του ξεναγού κινείται σύμφωνα με κώδικα που ορίζει η εφαρμογή και όχι σύμφωνα με εντολές του χρήστη. Συγκεκριμένα ο χαρακτήρας κινείται κάθε φορά πάνω στους κόμβους ενός μονοπατιού, η θέση του οποίου καθορίζεται κατά το στάδιο του σχεδιασμού του περιβάλλοντος. Τα μονοπάτια αποτελούν στιγμιότυπα της κλάσης Path ενώ οι κόμβοι τους αποτελούν στιγμιότυπα της κλάσης Marker.

Το datablock τύπου PlayerData που καθορίζει την μορφή του χαρακτήρα ονομάζεται GuideShape και επεκτείνει το datablock PlayerShape που περιγράφηκε παραπάνω. Η μορφή του ξεναγού προσδιορίζεται από το αρχείο Museum/data/shapes/guide/guide.dts όπως ορίζει η τιμή της παραμέτρου shapeFile του datablock.

Το datablock GuideShape υλοποιεί τις μεθόδους onReachDestination, onEndOfPath και onEndSequence που καλούνται όταν ο χαρακτήρας φτάσει στο σημείο που έχει οριστεί ως σημείο στόχος, όταν φτάσει τον τελευταίο κόμβο ενός μονοπατιού ή όταν ολοκληρώσει την εκτέλεση μία ακολουθίας κίνησης, αντίστοιχα.

Το datablock τύπου TSShapeConstructor με όνομα GuideDts αντιστοιχεί τα

εκτελέσιμα αρχεία guide_root.dsq, guide_forward.dsq, guide_back.dsq, guide_side.dsq του φακέλου Museum\data\shapes\guide στις κινήσεις root, run, back και side που αναγνωρίζει το Torque, αντίστοιχα

Ο χαρακτήρας του ξεναγού αποτελεί στιγμιότυπο της κλάσης AIPlayer και την κατασκευή του αναλαμβάνει ένα αντικείμενο AIManager όταν δοθεί η κατάλληλη εντολή από τον χρήστη. Η κλάση AIPlayer υλοποιεί ένα πλήθος μεθόδων που αφορούν στην κατασκευή, την κίνηση και γενικότερα την συμπεριφορά των στιγμιότύπων της. Οι κυριότερες μέθοδοι που χρησιμοποιούνται από το στιγμιότυπο του ξεναγού είναι οι spawn, spawnOnPath, followPath, moveToNode, moveToNextNode και done .

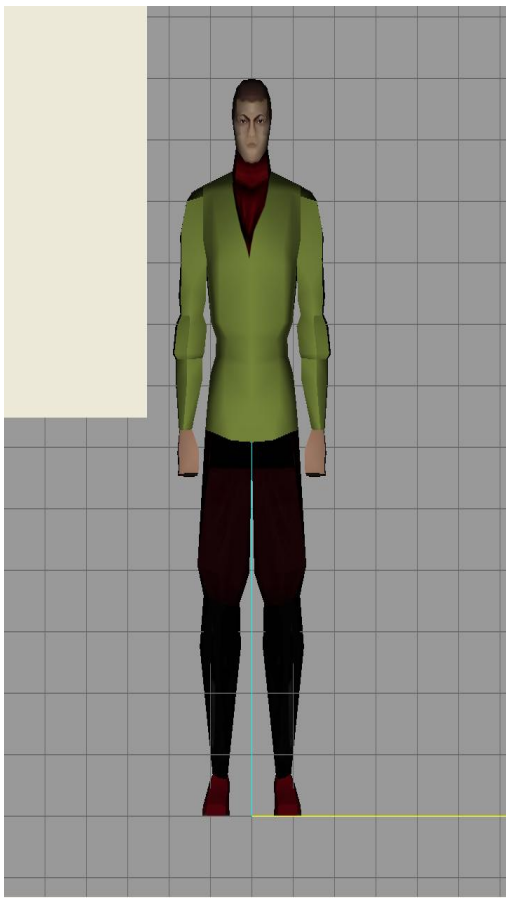
Η μέθοδος spawn κατασκευάζει ένα αντικείμενο της κλάσης AIPlayer, του δίνει το όνομα που καθορίζεται από το πρώτο όρισμά της και το τοποθετεί στο σημείο που καθορίζεται από το δεύτερο όρισμά της. Η spawnOnPath κατασκευάζει ένα αντικείμενο AIPlayer και το τοποθετεί στον πρώτο κόμβο του μονοπατιού που δέχεται ως όρισμα. Η followPath αναλαμβάνει την κίνηση του χαρακτήρα πάνω σε ένα συγκεκριμένο μονοπάτι. Η moveToNode μετακινεί τον χαρακτήρα σε έναν ορισμένο κόμβο του μονοπατιού. Η moveToNextNode μετακινεί τον χαρακτήρα στον επόμενο κόμβο του μονοπατιού, ενώ η done χρησιμοποιείται για την καταστροφή ενός αντικείμενο AIPlayer

Κάθε φορά που ο χρήστης ζητάει την εμφάνιση του ξεναγού, προκειμένου να έχει πληροφόρηση σχετικά με κάποιο έκθεμα, εμφανίζεται ο ξεναγός και κινείται προς το έκθεμα αυτό. Όταν φτάσει δίπλα στο έκθεμα σταματάει και εμφανίζονται οι αντίστοιχες πληροφορίες . Μετά από λίγα δευτερόλεπτα οι πληροφορίες εξαφανίζονται και ο ξεναγός απομακρύνεται από το έκθεμα.

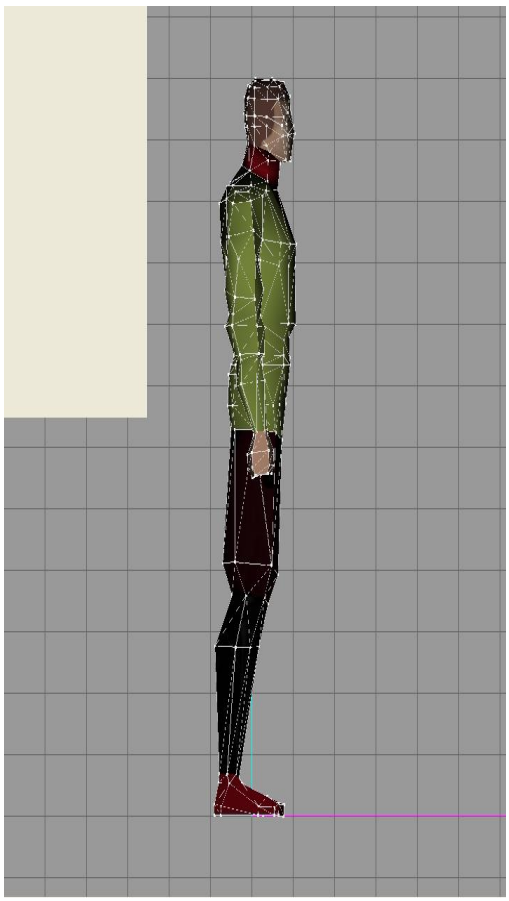
Συγκεκριμένα, όταν ο χρήστης ζητήσει την βοήθεια του ξεναγού, τότε κατασκευάζεται ένα αντικείμενο της κλάσης AIManager. Το αντικείμενο αυτό με την σειρά του κατασκευάζει ένα στιγμιότυπο της κλάσης AIPlayer καλώντας την μέθοδο spawnOnPath με όρισμα το μονοπάτι που πρέπει να ακολουθήσει ο χαρακτήρας. Η spawnOnPath απομονώνει τον πρώτο κόμβο του μονοπατιού και καλεί την μέθοδο spawn με όρισμα τον κόμβο αυτό. Η spawn αναλαμβάνει την δημιουργία του χαρακτήρα του ξεναγού και την εμφάνισή του στο σημείο όπου βρίσκεται ο κόμβος. Το αντικείμενο AIManager αφού κατασκευάσει τον χαρακτήρα καλεί την μέθοδο followPath που αναγκάζει τον χαρακτήρα να κινηθεί πάνω στο μονοπάτι που δέχεται σαν όρισμα η μέθοδος. Για την κίνηση του χαρακτήρα πάνω στο μονοπάτι χρησιμοποιούνται δύο ακόμα μέθοδοι: η moveToNextNode που αναλαμβάνει την κίνηση του χαρακτήρα στον επόμενο κόμβο του μονοπατιού και η moveToNode που αναλαμβάνει την κίνηση του χαρακτήρα στον κόμβο που δέχεται σαν όρισμα. Ο ξεναγός σταματάει μόλις φτάσει στον τελικό κόμβο του μονοπατιού που βρίσκεται δίπλα στο έκθεμα για το οποίο ενδιαφέρεται ο επισκέπτης. Στον χώρο αυτό υπάρχει ένα χωρικό trigger με αποτέλεσμα να καλείται η μέθοδος onEnterTrigger κατά την είσοδο του ξεναγού σε αυτό. Η μέθοδος αυτή ελέγχει αν ο χαρακτήρας που εισήλθε στον χώρο του trigger είναι ο ξεναγός και στην περίπτωση αυτή κάνει ορατό το αντικείμενο ObjectTriggerControler. Πριν την εμφάνιση του αντικειμένου αυτού που αποτελεί στιγμιότυπο της κλάσης GuiTextEditCtrl καθορίζεται το μήνυμα κειμένου που θα εμφανιστεί θέτοντας την παράμετρο text του αντικειμένου στην κατάλληλη τιμή. Επίσης καλείται η συνάρτηση moreInfo που κάνει ορατά τα αντικείμενα ArtistInfoTriggerControler και ExhibitInfoTriggerControler που εμφανίζουν πληροφορίες για τον καλλιτέχνη και για το έργο του αντίστοιχα. Τα μηνύματα αυτά παραμένουν ορατά για λίγα δευτερόλεπτα και στην συνέχεια εξαφανίζονται. Μόλις εξαφανιστούν δίνεται εντολή στον ξεναγό

να κινηθεί προς τον αρχικό κόμβο του μονοπατιού και στην συνέχεια εξαφανίζεται με κλήση της done για να εμφανιστεί ξανά όταν ο χρήστης χρειαστεί την βοήθειά του.

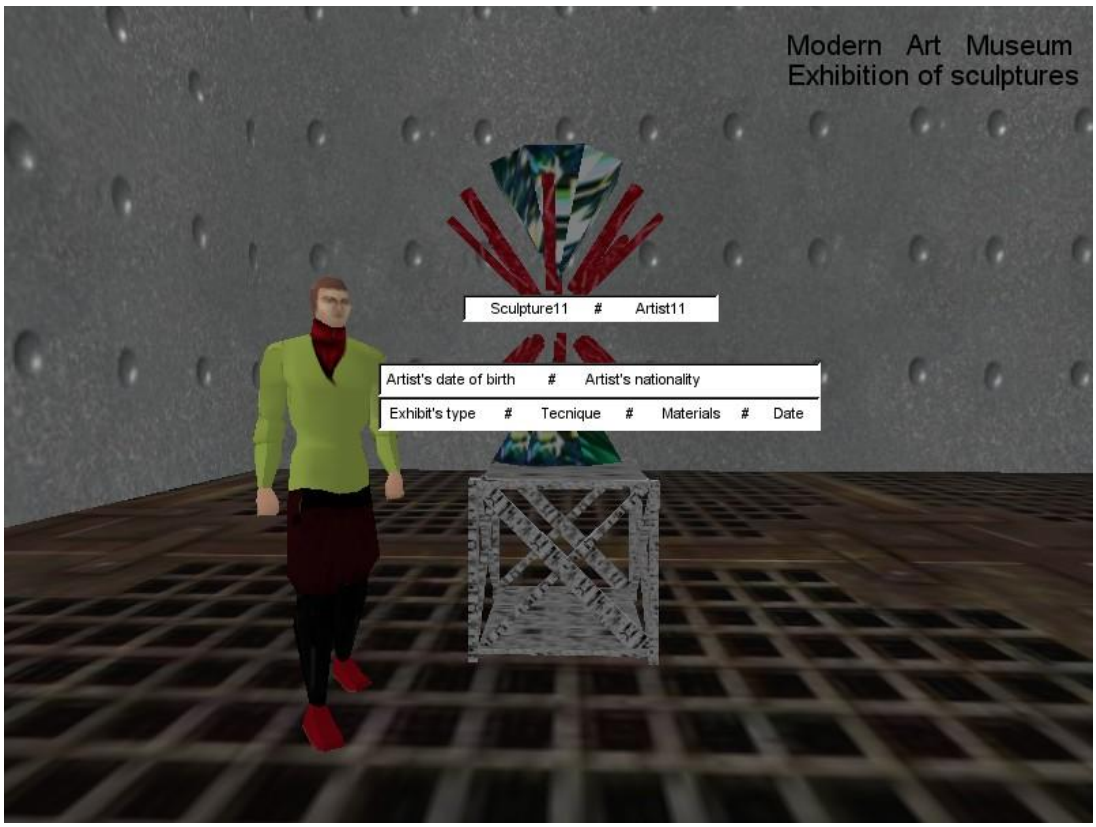
Το αρχείο dts που καθορίζει την μορφή του χαρακτήρα καθώς και τα εκτελέσιμα αρχεία dsq που καθορίζουν την κίνηση του βρίσκονται στον φάκελο Museum\data\shapes\ guide. Στον ίδιο φάκελο βρίσκονται αποθηκευμένα και τα textures
που χρησιμοποιεί το αρχείο dts για την διαμόρφωση της τελικής μορφής του χαρακτήρα.



Εικόνα 5.4 Στιγμιότυπο από την μοντελοποίηση του χαρακτήρα του ξεναγού στο Milkshape.



Εικόνα 5.5 Στιγμιότυπο από την μοντελοποίηση του χαρακτήρα του ξεναγού στο Milkshape.



Εικόνα 5.6 Ο χαρακτήρας του ξεναγού στο περιβάλλον του εικονικού κόσμου.

Ο κώδικάς που υλοποιεί τα datablocks και τις συναρτήσεις που χρησιμοποιεί ο χαρακτήρας του ξεναγού βρίσκεται στο αρχείο Museum\server\guide και παρατίθεται στην συνέχεια

```
//-----------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\server\guide
//-----------------------------------------------------------------------


//-----------------------------------------------------------------------
// Demo Pathed AIPlayer.
//-----------------------------------------------------------------------


datablock TSShapeConstructor(GuideDts)
{
   baseShape = "~/data/shapes/guide/guide.dts";
   sequence0 = "~/data/shapes/guide/guide_root.dsq root";
   sequence1 = "~/data/shapes/guide/guide_forward.dsq run";
   sequence2 = "~/data/shapes/guide/guide_back.dsq back";
   sequence3 = "~/data/shapes/guide/guide_side.dsq side";

};
```

```
datablock PlayerData(DemoPlayer : PlayerShape)
{
   shootingDelay = 2000;
   renderFirstPerson = false;
   shapeFile = "~/data/shapes/guide/guide.dts";

};

function DemoPlayer::onReachDestination(%this,%obj)
{
   // Moves to the next node on the path.

      if (%obj.currentNode == %obj.targetNode)
         %this.onEndOfPath(%obj,%obj.path);
      else
         %obj.moveToNextNode();

}

function DemoPlayer::onEndOfPath(%this,%obj,%path)
{
   echo("onEndOfPath");
   %obj.nextTask();
}

function DemoPlayer::onEndSequence(%this,%obj,%slot)
{
   echo("Sequence Done");
   %obj.stopThread(%slot);
   %obj.nextTask();
}


//--------------------------------------------------------------------------
// AIPlayer static functions
//--------------------------------------------------------------------------


function AIPlayer::spawn(%name,%spawnPoint)
{
   // Create the demo player object
    echo("spawn");
   %player = new AIPlayer() {
      dataBlock = DemoPlayer;
      path = "";
   };
   MissionCleanup.add(%player);
   %player.setShapeName(%name);
   %player.setTransform(%spawnPoint);

   return %player;
}


function AIPlayer::spawnOnPath(%name,%path)
{

   // Spawn a player and place him on the first node of the path
   echo("spawnOnPath");
   if (!isObject(%path))
      return;
```

```
   %node = %path.getObject(0);
   %player = AIPlayer::spawn(%name,%node.getTransform());
    return %player;
}

//-----------------------------------------------------------------------
// AIPlayer methods
//-----------------------------------------------------------------------

function AIPlayer::followPath(%this,%path,%node)
{
    echo("followPath");
   // Start the player following a path
if (!isObject(%this)) {
      %this=spawnOnPath("kk",%path);
   }

   %this.stopThread(0);
   if (!isObject(%path)) {
      %this.path = "";
      return;
   }

  %this.targetNode = %path.getCount() - 1;

  %this.path = %path;
  %this.moveToNode(1);


}

function AIPlayer::moveToNextNode(%this)
{
   echo("moveToNextNode" );
   if (%this.targetNode < 0 || %this.currentNode < %this.targetNode)
{
      if (%this.currentNode < %this.path.getCount() - 1)
         %this.moveToNode(%this.currentNode + 1);
      else
      {
        echo("return");
        return;
      }

   }
   else
      if (%this.currentNode == 0)
    {
      %this.moveToNode(%this.path.getCount() - 1);
      echo("currentNode == 0");
      }
      else
      {
         %this.moveToNode(%this.currentNode - 1);
         echo("moveToNode%this.currentNode - 1");
      }
}
```

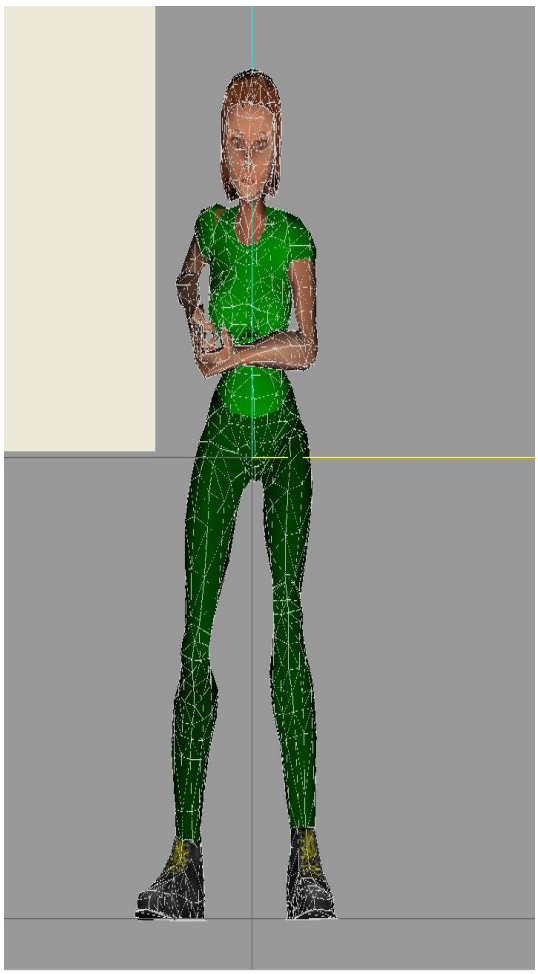
```
function AIPlayer::moveToNode(%this,%index)
{
   echo("moveToNode" @ " " @ %this.path @ " " @ %index);
   // Move to the given path node index
   %this.currentNode = %index;
   %node = %this.path.getObject(%index);
   %this.setMoveDestination(%node.getTransform(), %index ==
%this.targetNode);
}


//-----------------------------------------------------------------------

function AIPlayer::pushTask(%this,%method)
{
   if (%this.taskIndex $= "") {
      %this.taskIndex = 0;
      %this.taskCurrent = -1;
   }
   %this.task[%this.taskIndex] = %method;
   %this.taskIndex++;
   if (%this.taskCurrent == -1)
      %this.executeTask(%this.taskIndex - 1);
}

function AIPlayer::clearTasks(%this)
{
   %this.taskIndex = 0;
   %this.taskCurrent = -1;
}

function AIPlayer::nextTask(%this)
{
   if (%this.taskCurrent != -1)
      if (%this.taskCurrent < %this.taskIndex - 1)
         %this.executeTask(%this.taskCurrent++);
      else
         %this.taskCurrent = -1;
}

function AIPlayer::executeTask(%this,%index)
{
   %this.taskCurrent = %index;
   eval(%this.getId() @ "." @ %this.task[%index] @ ";");
}


function AIPlayer::wait(%this,%time)
{
   %this.schedule(%time * 1000,"nextTask");
}

function AIPlayer::done(%this,%time)
{
   %this.schedule(0,"delete");
}
```

```
function AIPlayer::animate(%this,%seq)
{
   %this.stopThread(0);
   //%this.playThread(0,%seq);
   //%this.setActionThread(%seq);
}


//--------------------------------------------------------------------


function AIManager::think(%this,%path)
{

   // We could hook into the player's onDestroyed state instead of
   // having to "think", but thinking allows us to consider other
   // things...
   if (!isObject(%this.player))
      %this.player = %this.spawn(%path);
   %this.schedule(500,think);

}



function AIManager::spawn(%this, %choosepath)
{

   %player = AIPlayer::spawnOnPath("Kork",%choosepath);
   %player.followPath(%choosepath,-1);
   %player.schedule(10000,"delete");
   //%player.mountImage(CrossbowImage,0);
   //%player.setInventory(CrossbowAmmo,1000);
   return %player;
}
//--------------------------------------------------------------------
```

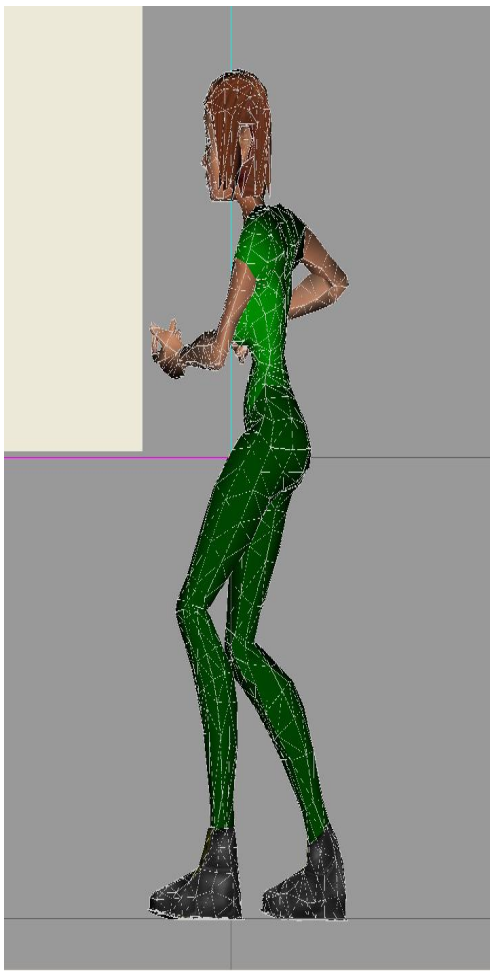
5.3 Ο χαρακτήρας του υπαλλήλου του μουσείου

Ο χαρακτήρας του υπαλλήλου του μουσείου που βρίσκεται στο γραφείο πληροφοριών διαφέρει από τους χαρακτήρες που περιγράφηκαν προηγουμένως ως προς το ότι δεν είναι απαραίτητη η κίνησή του κατά την εκτέλεση της εφαρμογής. Για τον λόγο αυτό δεν έχει σχεδιαστεί animation για τον χαρακτήρα αυτό και έχουν ακολουθηθεί μόνο οι διαδικασίες της μοντελοποίησης και του texturing κατά τον σχεδιασμό του.

Ο χαρακτήρας υλοποιείται σαν στατική μορφή δηλαδή σαν στιγμιότυπο της κλάσης StaticShape και εισάγεται εξ' αρχής στον εικονικό κόσμο κατά το στάδιο του σχεδιασμού του περιβάλλοντος. Ο χαρακτήρας του γραφείου πληροφοριών χρησιμοποιεί το datablock Guide που ορίζει την μορφή του χαρακτήρα. Συγκεκριμένα, όπως υποδηλώνει η τιμή της παραμέτρου shapeFile το αρχείο dts που καθορίζει την μορφή του χαρακτήρα είναι το αρχείο player που βρίσκεται αποθηκευμένο στον φάκελο museumEmp. Στον ίδιο φάκελο περιέχονται και τα texture που χρησιμοποιεί ο χαρακτήρας του υπαλλήλου του μουσείου.



Εικόνα 5.7 Στιγμιότυπο από την μοντελοποίηση του χαρακτήρα της υπαλλήλου στο Milkshape.



Εικόνα 5.8 Στιγμιότυπο από την μοντελοποίηση του χαρακτήρα της υπαλλήλου στο Milkshape.

Εικόνα 5.9 Ο χαρακτήρας της υπαλλήλου του γραφείου πληροφοριών στο περιβάλλον του εικονικού κόσμου.

Ο κώδικάς του αρχείου Museum\data\missions\MuseumMission που κατασκευάζει το αντικείμενο που αντιστοιχεί στο μοντέλο του χαρακτήρα καθώς και το datablock Guide που βρίσκεται στο αρχείο Museum\server\museumEmp και καθορίζει την μορφή του χαρακτήρα παρατίθενται στην συνέχεια.

```
//--------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\data\missions\MuseumMission
//--------------------------------------------------------------------


new StaticShape(guide1) {
      position = "42 -312 2.6";
      rotation = "0 0 1 180";
      scale = "1 1 1";
      dataBlock = "Guide";
   };



//--------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\server\museumEmp
//--------------------------------------------------------------------
```

```
datablock StaticShapeData(Guide)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/museumEmp /player.dts";
};

.//--------------------------------------------------------------------------------------------
```

6. ΑΛΛΗΛΕΠΙΔΡΑΣΗ ΧΡΗΣΤΗ-ΕΠΙΣΚΕΠΤΗ ΜΕ ΤΗΝ ΕΦΑΡΜΟΓΗ ΚΑΙ ΤΟ ΠΕΡΙΒΑΛΛΟΝ ΤΟΥ ΕΙΚΟΝΙΚΟΥ ΚΟΣΜΟΥ.

Κύριο χαρακτηριστικό ενός μεγάλου πλήθους σύγχρονων εφαρμογών πολυμέσων είναι η δυνατότητα που παρέχουν στον χρήστη να αλληλεπιδρά με την εφαρμογή. Σκοπός της αλληλεπιδραστικότητας(interactivity) είναι η προσαρμογή της εφαρμογής στις ανάγκες του κάθε χρήστη δίνοντας του την δυνατότητα να καθορίζει ορισμένα χαρακτηριστικά του τρόπου παρουσίασης της παρεχόμενης πληροφορίας. Μια interactive εφαρμογή μπορεί να περιέχει τρεις βαθμούς προσαρμοστικότητας στις επιθυμίες του χρήστη, που αφορούν στον καθορισμό της σειράς, της ταχύτητας και της μορφής παρουσίασης της πληροφορίας. Κάθε εφαρμογή που υποστηρίζει τουλάχιστον μία από τις παραπάνω δυνατότητες, περιέχει ένα αυτόματο σύστημα παρουσίασης της πληροφορίας που δέχεται τις εντολές του χρήστη, με αποτέλεσμα την μη-γραμμική παρουσίαση της πληροφορίας. Αντίθετα σε εφαρμογές που υποστηρίζουν γραμμική ή παθητική παρουσίαση της πληροφορίας, ακολουθείται ένα προκαθορισμένο σχέδιο παρουσίασης πάνω στο οποίο ο χρήστης δεν έχει κανένα ουσιαστικό έλεγχο ή δυνατότητα παρέμβασης.

Στην εφαρμογή της πλοήγησης στον εικονικό κόσμο του μουσείου, πέραν της σχεδίασης του περιβάλλοντος, εξίσου σημαντικό είναι να δοθεί η δυνατότητα στον χρήστη να επισκέπτεται όποια τοποθεσία του εικονικού κόσμου επιθυμεί και να αλληλεπιδρά με τα αντικείμενα που τον περιβάλλουν.

Συγκεκριμένα ο χρήστης έχει την δυνατότητα να κινεί τον χαρακτήρα του επισκέπτη προς όποια κατεύθυνση επιθυμεί, να επισκέπτεται οποιοδήποτε μέρος του εικονικού κόσμου και να επιλέγει τα κτίρια και τις εκθέσεις που θέλει να επισκεφθεί με όποια σειρά επιλέξει. Οδηγίες για την κίνηση του χαρακτήρα του επισκέπτη εμφανίζονται στην οθόνη με πάτημα συγκεκριμένου πλήκτρου που καθορίζεται από την εφαρμογή.

Επίσης ο χρήστης έχει την δυνατότητα να ζητάει και να του παρέχονται πληροφορίες σχετικά με τις εκθέσεις του μουσείου ή σχετικά με συγκεκριμένα εκθέματα όταν το θεωρεί απαραίτητο. Επιπλέον, όταν επιθυμεί περαιτέρω ενημέρωση σχετικά με κάποιο έκθεμα μπορεί να επιλέξει την εμφάνιση του ξεναγού ο οποίος αναλαμβάνει την αναλυτικότερη περιγραφή του εκθέματος. Τέλος, για την διευκόλυνση του χρήστη είναι απαραίτητη η εμφάνιση παραθύρων με ενημερωτικά μηνύματα κατά την είσοδό του σε συγκεκριμένους χώρους και κατά την έναρξη της εφαρμογής.

Γενικά είναι σημαντικό ο επισκέπτης να μπορεί να αλληλεπιδρά με το περιβάλλον και να μπορεί να λαμβάνει αποφάσεις που επηρεάζουν την ροή εκτέλεσης της εφαρμογής. Η εφαρμογή περιλαμβάνει δύο βαθμούς προσαρμοστικότητας στις επιθυμίες του χρήστη που αφορούν στην σειρά και μορφή παρουσίασης της πληροφορίας.

Η αλληλεπίδραση στην εφαρμογή του εικονικού κόσμου του μουσείου, υλοποιείται κυρίως τους τρεις τρόπους που παρατίθενται παρακάτω:

(α) Με το binding που επιτρέπει την αντιστοίχηση ορισμένων πλήκτρων σε ορισμένες ενέργειες. Με τον τρόπο αυτό παρέχονται στον χρήστη δυνατότητες επιλογών, με το πάτημα των αντίστοιχων πλήκτρων.

(β) Με τα χωρικά triggers (area triggers) που οριοθετούν συγκεκριμένες περιοχές του εικονικού κόσμου. Στην συνέχεια με υλοποίηση των κατάλληλων μεθόδων των

triggers καθορίζονται οι διαδικασίες που ενεργοποιούνται όταν ο επισκέπτης βρίσκεται στις οριοθετημένες περιοχές και όταν εισέρχεται ή εξέρχεται από αυτές.
(γ) Μέσω των callback συναρτήσεων και μεθόδων που υλοποιούν τα interactive αντικείμενα του περιβάλλοντος και καθορίζουν την συμπεριφορά των αντικειμένων σε συγκεκριμένες περιστάσεις. Παράδειγμα τέτοιας περίστασης αποτελεί η σύγκρουση του χαρακτήρα με αντικείμενα που υλοποιούν την μέθοδο onCollision, η οποία καθορίζει την συμπεριφορά του αντικειμένου κατά την σύγκρουση.

6.1 Αντιστοίχηση πλήκτρων σε συγκεκριμένες ενέργειες (binding)

Το binding πραγματοποιείται με κλήση της μεθόδου bind του αντικειμένου moveMap που αποτελεί στιγμιότυπο της κλάσης ActionMap. Η κατασκευή του αντικειμένου και ο ορισμός των μεθόδων του βρίσκονται στο αρχείο Museum\client\defaultBind.

Η βασικότερη χρήση του binding είναι η δυνατότητα που παρέχει στο χρήστη να κινεί τον χαρακτήρα προς όποια κατεύθυνση επιθυμεί, αποφασίζοντας με αυτό τον τρόπο την πορεία που θα ακολουθήσει ο επισκέπτης κατά την πλοήγησή του στον εικονικό κόσμο. Τα πλήκτρα που χρησιμοποιούνται για την κίνηση του χαρακτήρα, οι επιτρεπτές κινήσεις του και οι συναρτήσεις που καλούνται για την πραγματοποίηση της κάθε κίνησης φαίνονται στον παρακάτω πίνακα.

Α/Α	**Πλήκτρο**	**Κίνηση χαρακτήρα**	**Συνάρτηση που υλοποιεί την κίνηση**
1	w	Κίνηση προς τα μπρος	moveforward
2	s	Κίνηση προς τα πίσω	monebackward
3	a	Κίνηση προς τα αριστερά	moveleft
4	d	Κίνηση προς τα δεξιά	moveright
5	space	Άλμα	jump

Πίνακας 6.1 Binding για την κίνηση του επισκέπτη.

Εκτός των πλήκτρων που χρησιμοποιούνται για την κίνηση του χαρακτήρα προς την επιθυμητή κατεύθυνση, τα πλήκτρα ή συνδυασμοί πλήκτρων που παρουσιάζονται στη συνέχεια, δίνουν την δυνατότητα στον χρήστη να κάνει επιλογές που σχετίζονται με τον τρόπο παρουσίασης της εφαρμογής και την εμφάνιση παραθύρων που περιέχουν μηνύματα πληροφοριών.

Έτσι το πάτημα του πλήκτρου escape εμφανίζει παράθυρο διαλόγου που καλεί τον χρήστη να επιλέξει αν θέλει να συνεχίσει την πλοήγηση στον εικονικό κόσμο ή να την εγκαταλείψει. Αν ο χρήστης επιλέξει έξοδο από τον εικονικό κόσμο, τότε καλείται η συνάρτηση exit που αντιστοιχεί σε επιστροφή στο αρχικό μενού της εφαρμογής. Αν επιλέξει συνέχιση της πλοήγησης στον εικονικό κόσμο το παράθυρο διαλόγου εξαφανίζεται χωρίς να εκτελεστεί καμία περαιτέρω ενέργεια.

Το πλήκτρο tab δίνει την δυνατότητα στον χρήστη να επιλέξει την εμφάνιση ή απόκρυψη του χαρακτήρα του επισκέπτη. Το πλήκτρο αυτό ουσιαστικά καθορίζει αν η κάμερα θα βρίσκεται πίσω η μπροστά από τον χαρακτήρα του επισκέπτη.

Το πλήκτρο h χρησιμοποιείται για την εμφάνιση παραθύρου που εμφανίζει πληροφορίες για την κίνηση του επισκέπτη. Συγκεκριμένα τον ενημερώνει ποια

πλήκτρα πρέπει να χρησιμοποιήσει για την κίνηση του επισκέπτη προς την κάθε κατεύθυνση και το πλήκτρο που αντιστοιχεί σε έξοδο από το περιβάλλον του εικονικού κόσμου και επιστροφή στο αρχικό μενού της εφαρμογή.

Το πλήκτρο tilde δίνει την δυνατότητα εμφάνισης ή απόκρυψης του παραθύρου της κονσόλας. Το interface της κονσόλας επιτρέπει την επισκόπηση των ενεργειών που έχουν εκτελεστεί και την εισαγωγή εντολών για άμεση εκτέλεση.

Ο συνδυασμός πλήκτρων alt+enter δίνει την δυνατότητα επιλογής για την παρουσίαση της εφαρμογής σε παράθυρο που καλύπτει ολόκληρη την οθόνη ή σε μικρότερο παράθυρο, ανάλογα με την επιθυμία του χρήστη.

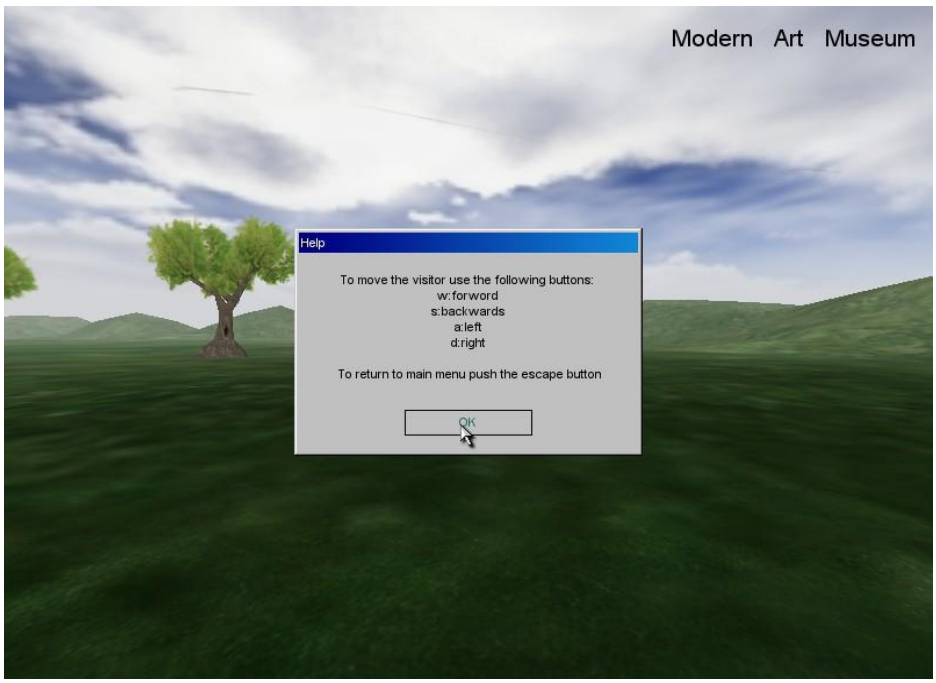
Τέλος ο συνδυασμός πλήκτρων alt+c παρέχει στον χρήστη τη δυνατότητα να αλλάξει την λήψη της κάμερας. Αρχικά η κάμερα ακολουθεί τον χαρακτήρα του επισκέπτη ενώ με χρήση των πλήκτρων alt και c η κάμερα απομακρύνεται από τον επισκέπτη και η λήψη της γίνεται πανοραμική. Στην περίπτωση αυτή τα πλήκτρα που αφορούν στην κίνηση του χαρακτήρα, χρησιμοποιούνται για την κίνηση της κάμερας, ενώ ο χαρακτήρας του επισκέπτη παραμένει ακίνητος. Για να ξαναποκτήσει ο χρήστης τον έλεγχο της κίνησης του χαρακτήρα πρέπει να επαναφέρει την κάμερα στη θέση του επισκέπτη με νέο πάτημα του συνδυασμού πλήκτρων alt+c.

Η κίνηση του ποντικιού αντιστοιχεί σε κάθετη ή οριζόντια περιστροφή της κάμερας και ταυτόχρονη περιστροφή του χαρακτήρα σε περίπτωση που η λήψη της κάμερας δεν είναι πανοραμική. Η περιστροφή αυτή πραγματοποιείται με κλήση των συναρτήσεων yaw και pitch, που υπολογίζουν την γωνία περιστροφής που αντιστοιχεί σε κάθε κίνηση του ποντικιού.

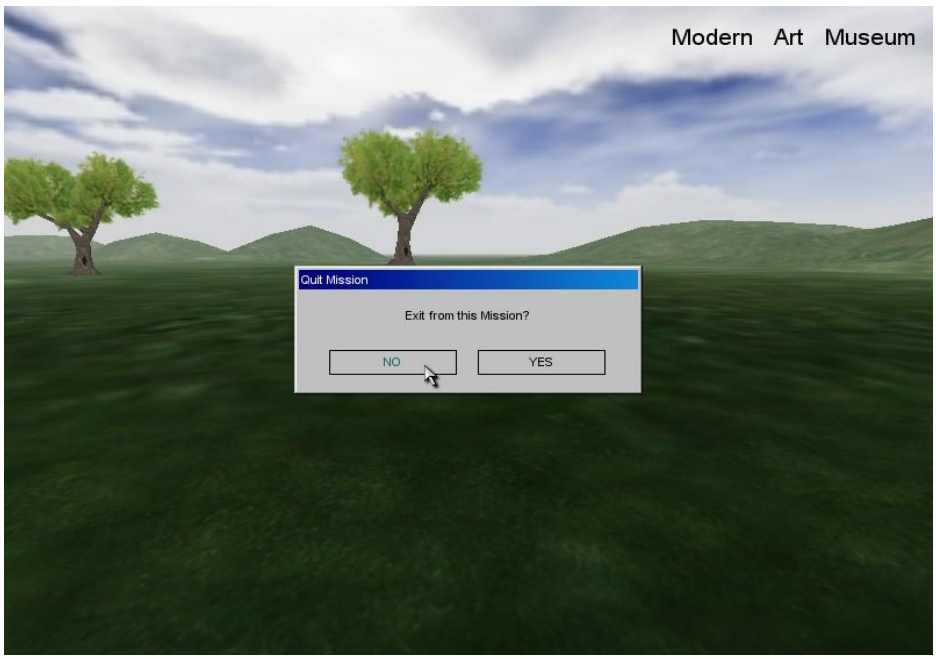
Τα πλήκτρα και οι συνδυασμοί πλήκτρων που περιγράφηκαν παραπάνω, οι ενέργειες που αντιστοιχούν σε αυτά και οι συναρτήσεις που καλούνται κατά το πάτημα τους φαίνονται στον παρακάτω πίνακα.

Α/Α	**Πλήκτρο**	**Εκτελούμενη ενέργεια**	**Καλούμενη συνάρτηση**
1	escape	Έξοδος από τον εικονικό κόσμο	escapeFromGame
2	tab	Εμφάνιση ή απόκρυψη χαρακτήρα επισκέπτη	toggleFirstPerson
3	h	Εμφάνιση παραθύρου πληροφοριών	displayHelpMessage
4	tidle	Εμφάνιση ή απόκρυψη παραθύρου κονσόλας	toggleConsole
5	alt+enter	Επιλογή μεγέθους παραθύρου εφαρμογής	toggleFullScreen
6	alt+c	Αλλαγή λήψης της κάμερας	toggleCamera

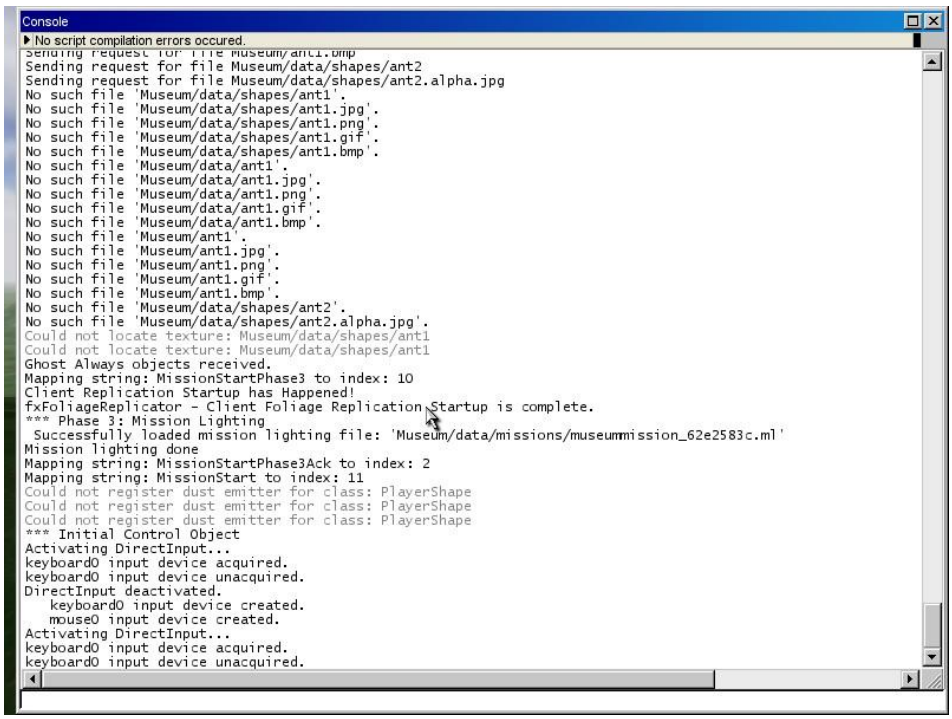
Πίνακας 6.2 Αντιστοίχηση πλήκτρων σε συγκεκριμένες ενέργειες.



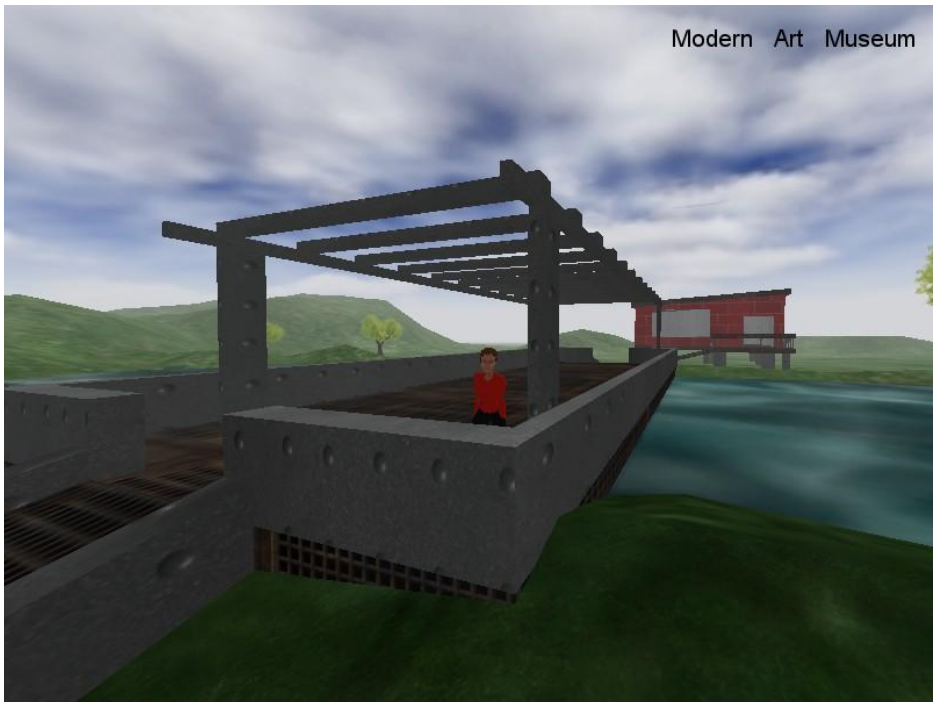
Εικόνα 6.1 Παράθυρο βοήθειας που εμφανίζεται με πάτημα του πλήκτρου 'H'



Εικόνα 6.2 Παράθυρο διαλόγου που εμφανίζεται με πάτημα του πλήκτρου 'escape'



Εικόνα 6.3 Το interface της κονσόλας εμφανίζεται με πάτημα του πλήκτρου 'tidle'



Εικόνα 6.4 Πανοραμική λήψη της κάμερας με πάτημα του συνδυασμού πλήκτρων alt+c

Οι συναρτήσεις και μέθοδοι που πραγματοποιούν το binding και αυτές που εκτελούνται κατά το πάτημα των αντίστοιχων πλήκτρων βρίσκονται αποθηκευμένες στο αρχείο Museum\client\defaultBind, τα περιεχόμενα του οποίου παρουσιάζονται στην παράγραφο 8.2.4 του παραρτήματος.

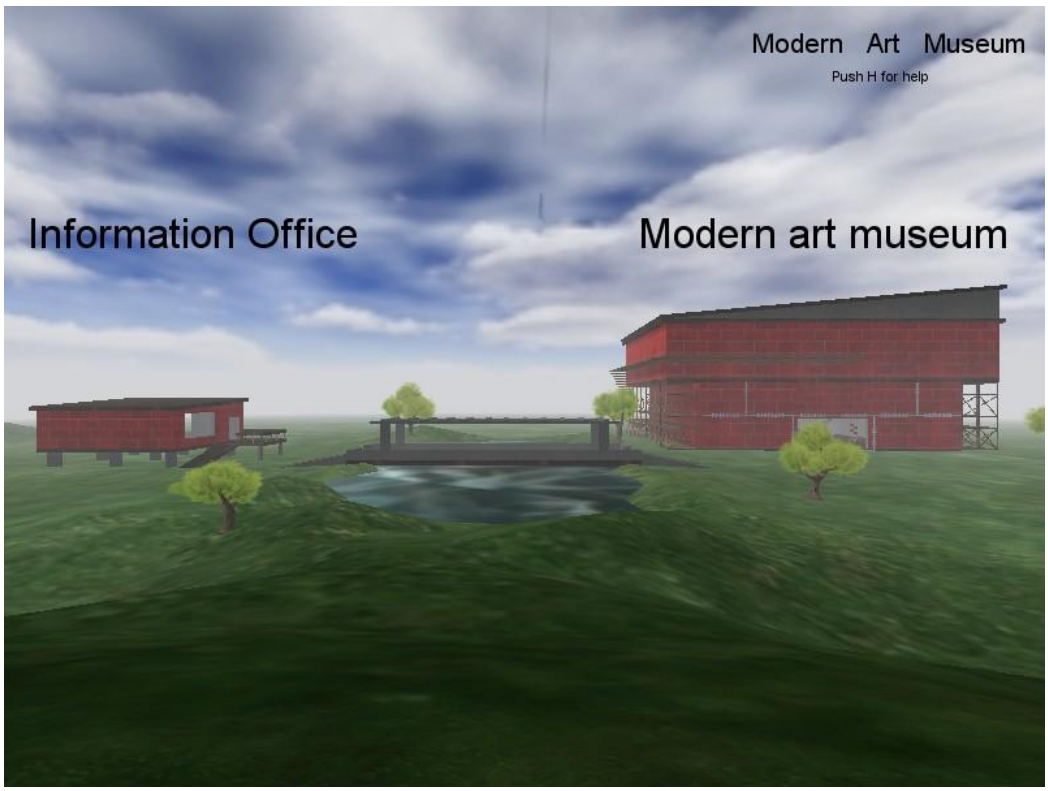
6.2 Αλληλεπίδραση μέσω χωρικών triggers (area triggers)

Τα χωρικά triggers είναι στιγμιότυπα της κλάσης Trigger. Πρόκειται για αντικείμενα που οριοθετούν περιοχές του εικονικού κόσμου στις οποίες προσδίδουν συγκεκριμένες ιδιότητες. Το είδος του κάθε trigger καθορίζεται από το datablock που χρησιμοποιεί και η συμπεριφορά του καθορίζεται από τις μεθόδους που υλοποιεί το datablock αυτό.

Οι κύριες μέθοδοι που υλοποιεί κάθε trigger είναι οι onEnterTrigger, onLeaveTrigger και onTickTrigger. Οι μέθοδοι αυτές καλούνται κάθε φορά που ένα αντικείμενο εισέρχεται, εξέρχεται ή παραμένει, αντίστοιχα, στον οριοθετημένο από το trigger χώρο. Η χρήση των triggers συμβάλλει στην υλοποίηση της αλληλεπίδρασης του επισκέπτη με το περιβάλλον του εικονικού κόσμου, αφού με την βοήθεια των triggers μπορεί να ελεγχθεί η είσοδος, έξοδος και παραμονή του επισκέπτη σε χώρους που περιβάλλονται από κάποιο trigger και να πραγματοποιηθεί η εκτέλεση συγκεκριμένων ενεργειών.

Τα triggers που χρησιμοποιούνται για την υλοποίηση της αλληλεπίδρασης του επισκέπτη είναι τα StartTrigger , ExpoOneTrigger, ExpoTwoTrigger και InfoHouseTrigger που οριοθετούν τέσσερις συγκεκριμένους χώρους του εικονικού κόσμου

Το StartTrigger ανήκει στην κατηγορία που καθορίζει το ομώνυμο datablock. Περιβάλλει τον χώρο γύρω από το σημείο εμφάνισης του χαρακτήρα κατά την εκκίνηση της πλοήγησης στον εικονικό κόσμο, δηλαδή γύρω από το αντικείμενο της κλάσης SpawnSphere . Το trigger αυτό αναλαμβάνει την εμφάνιση ενός μηνύματος GuiTextCtrl με όνομα StartTriggerControler που περιλαμβάνει πληροφορίες οι οποίες αφορούν στα κτίρια που βλέπει ο χαρακτήρας κατά την είσοδο του στον εικονικό κόσμο. Συγκεκριμένα πληροφορεί τον επισκέπτη ότι το κτίριο που βλέπει αριστερά είναι το γραφείο πληροφοριών ενώ το κτίριο που βρίσκεται στο δεξί μέρος της οθόνης είναι το μουσείο. Η εμφάνιση του GUI διαρκεί δύο δευτερόλεπτα και απενεργοποιείται με το πέρας του χρονικού αυτού διαστήματος. Το GUI εμφανίζεται κατά την είσοδο του χαρακτήρα στο trigger με overriding της μεθόδου onEnterTrigger. Επίσης αρχικοποιεί την παράμετρο visible του PlaceTriggerControler στην τιμή 0. Ενώ η έξοδος του χαρακτήρα από τον χώρο του trigger που αντιστοιχεί σε κλήση της μεθόδου onLeaveTrigger καθώς και η παραμονή του στο χώρο αυτό που αντιστοιχεί σε κλήση της συνάρτησης OnTickTrigger δεν ενεργοποιούν καμία διαδικασία, αφού έχουν υλοποιηθεί ως κενές μέθοδοι. Η μεταβλητή tickPeriodMS που τίθεται στην τιμή 100ms κατά την κατασκευή του datablock του StartTrigger καθορίζει το χρονικό διάστημα ανάμεσα σε δύο διαδοχικές κλήσεις της συνάρτησης OnTickTrigger σε περίπτωση που κάποιο αντικείμενο βρίσκεται στον χώρο του trigger.

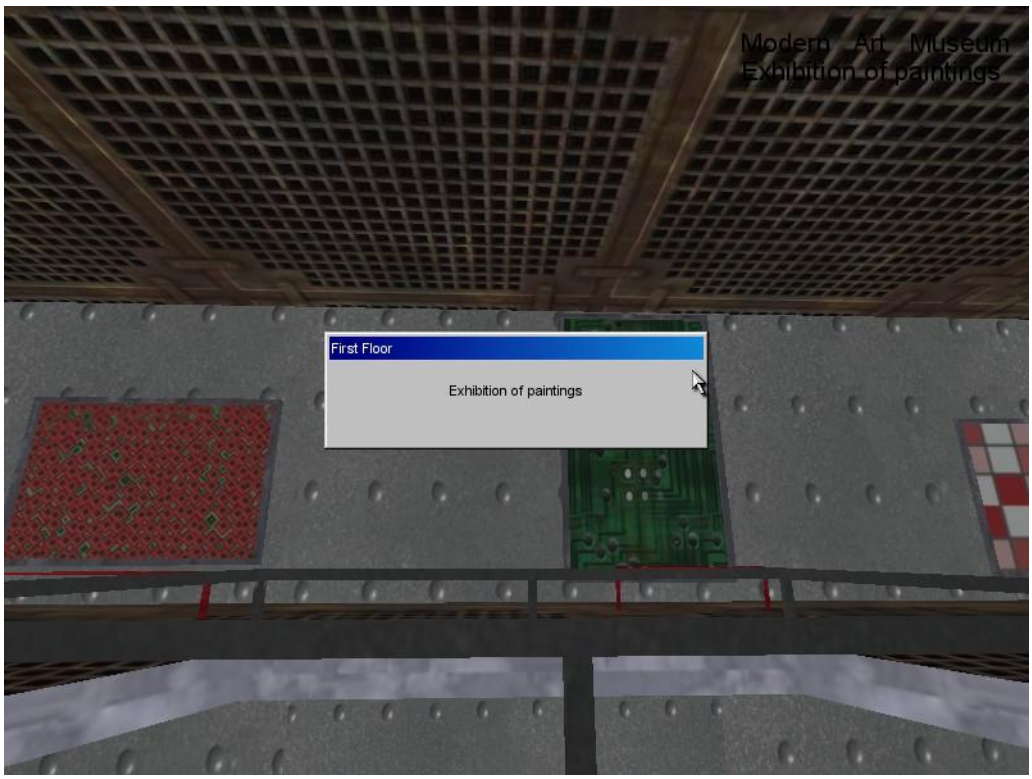


Εικόνα 6.5 Τα μηνύματα που εμφανίζονται από την μέθοδο onEnterTrigger του StartTrigger.

Το trigger με ονομασία DownFloorTrigger ανήκει στην κατηγορία triggers της οποίας οι ιδιότητες καθορίζονται από το datablock ExpoOneTrigger. Το DownFloorTrigger περιβάλλει τον χώρο του ισογείου του μουσείου όπου βρίσκονται τα εκθέματα της έκθεσης γλυπτών. Κατά την είσοδο κάποιου χαρακτήρα στον χώρο του trigger καλείται η μέθοδος onEnterTrigger που πραγματοποιεί έλεγχο για το αν ο χαρακτήρας αυτός είναι ο επισκέπτης του μουσείου. Στην περίπτωση αυτή εμφανίζεται παράθυρο διαλόγου MessagePopup με επικεφαλίδα "Ground Floor" και μήνυμα κειμένου "Exhibition of sculptures" το οποίο ενημερώνει τον επισκέπτη για την είσοδο του στο στην έκθεση γλυπτών του μουσείου και εξαφανίζεται μετά το πέρας 1,5 δευτερολέπτου. Καθ' όλη την διάρκεια παραμονής του επισκέπτη στον χώρο της έκθεσης γλυπτών είναι ορατό το αντικείμενο PlaceTriggerControler το οποίο αποτελεί στιγμιότυπο της κλάσης GuiTextCtrl αφού πρώτα μετατραπεί το μήνυμα που φέρει το αντικείμενο σε "Exhibition of sculptures". Το μήνυμα αυτό εμφανίζεται στην πάνω δεξιά γωνία της οθόνης και εξαφανίζεται κατά την έξοδο του επισκέπτη από τον χώρο του DownFloorTrigger. Αυτό επιτυγχάνεται με την κλήση της μεθόδου onLeaveTrigger που καλείται κάθε φορά που ο επισκέπτης εγκαταλείπει τον χώρο του trigger. Η μέθοδος αυτή θέτει την παράμετρο visible του PlaceTriggerControler στην τιμή 0. Η μέθοδος OnTickTrigger δεν ενεργοποιεί καμία διαδικασία, αφού έχει υλοποιηθεί ως κενή μέθοδος.

Εικόνα 6.6 Είσοδος στην έκθεση γλυπτών.

Ο πρώτος όροφος του μουσείου που περιέχει τα εκθέματα της έκθεσης πινάκων ζωγραφικής περιβάλλεται από το trigger με όνομα UpFloorTrigger το οποίο χρησομοποιεί το datablock ExpoTwoTrigger. Το datablock αυτό καθορίζει την τιμή της μεταβλητής tickPeriodMS που τίθεται στην τιμή 100ms όπως και στα προηγούμενα triggers. Αντίστοιχα με την περίπτωση του DownFloorTrigger. κατά την είσοδο κάποιου χαρακτήρα στον χώρο του trigger καλείται η μέθοδος onEnterTrigger που πραγματοποιεί έλεγχο για το αν ο χαρακτήρας αυτός είναι ο επισκέπτης του μουσείου. Στην περίπτωση αυτή εμφανίζεται παράθυρο διαλόγου MessagePopup με επικεφαλίδα " First floor" και μήνυμα κειμένου "Exhibition of paintings" το οποίο ενημερώνει τον επισκέπτη για την είσοδο του στο στην έκθεση πινάκων ζωγραφικής του μουσείου και εξαφανίζεται μετά το πέρας 1,5 δευτερολέπτου. Επίσης η μέθοδος onEnterTrigger του UpFloorTrigger κάνει ορατό το αντικείμενο PlaceTriggerControler αφού πρώτα μετατραπεί το μήνυμα που φέρει το αντικείμενο σε "Exhibition of paintings". Η κλήση της onLeaveTrigger κατά την έξοδο του επισκέπτη από τον χώρο του ExpoTwoTrigger εξαφανίζει το μήνυμα "Exhibition of paintings" από την οθόνη θέτοντας την παράμετρο visible του PlaceTriggerControler στην τιμή 0. Η συνάρτηση OnTickTrigger υλοποιείται ως κενή μέθοδος.



Εικόνα 6.7 Είσοδος στην έκθεση πινάκων ζωγραφικής.

Το trigger infoOfficeTrigger περιβάλλει το γραφείο πληροφοριών του μουσείο και χρησιμοποιεί datablock InfoHouseTrigger. Κατά την είσοδο του επισκέπτη στο γραφείο που ισοδυναμεί με κλήση της μεθόδου onEnterTrigger εμφανίζεται παράθυρο διαλόγου MessagePopup με επικεφαλίδα " Information office" και μήνυμα κειμένου "Welcome to the information office" το οποίο ενημερώνει τον επισκέπτη για την είσοδο του στο στην έκθεση πινάκων ζωγραφικής του μουσείου και εξαφανίζεται μετά από 1,5 δευτερόλεπτο. Καθόλη την διάρκεια της παραμονής του επισκέπτη στο χώρο του γραφείου είναι ορατό στην άνω δεξιά γωνία της οθόνης το μήνυμα "Information office", αφού η μέθοδος onEnterTrigger πρώτα μετατρέψει το μήνυμα που φέρει το αντικείμενο PlaceTriggerControler σε "Information office", και στη συνέχεια θέτει την παράμετρο visible στην τιμή 1. Το μήνυμα παύει να είναι ορατό όταν κατά την έξοδο του επισκέπτη από το χώρο του infoOfficeTrigger η παράμετρος visible του PlaceTriggerControler τίθεται εκ νέου στην τιμή 0.

Εικόνα 6.8 Είσοδος στο γραφείο πληροφοριών.

Ο κώδικας που αναφέρεται στην κατασκευή των trigger που περιγράφηκαν παραπάνω βρίσκεται στο αρχείο Museum\data\missions\MuseumMission, ενώ ο κώδικας που περιγράφει τα datablock των trigger και τις μεθόδους onEnterTrigger, onLeaveTrigger και onTickTrigger που υλοποιεί το καθένα βρίσκεται στο αρχείο Museum\server\triggers. Τα δύο αυτά αρχεία παρατίθενται στο παράρτημα του κεφαλαίου 8 στις παραγράφους 8.3.1 και 8.1.3, αντίστοιχα.

6.3 Αλληλεπίδραση μέσω συναρτήσεων και μεθόδων callback.

Οι συναρτήσεις και οι μέθοδοι callback αποτελούν μία ειδική κατηγορία συναρτήσεων και μεθόδων που καλούνται όταν πραγματοποιείται κάποιο συγκεκριμένο, προκαθορισμένο γεγονός. Παραδείγματα τέτοιων μεθόδων είναι οι μέθοδοι onAction, onCollision και onRemove. Η μέθοδος onAction καλείται από αντικείμενα που αποτελούν συστατικά στοιχεία των Graphical User Interfaces(GUI), όταν συμβεί κάποιο γεγονός που ελέγχεται από το συγκεκριμένο αντικείμενο που καλεί την μέθοδο. Η μέθοδος onCollision καλείται κατά τη σύγκρουση δύο αντικειμένων, ενώ η μέθοδος onRemove καλείται λίγο πριν την διαγραφή ενός αντικειμένου από τον εικονικό κόσμο. Στον εικονικό κόσμο του μουσείου έχει υλοποιηθεί η callback μέθοδος onCollision που καλείται όταν ο επισκέπτης έρθει σε επαφή με συγκεκριμένα interactive αντικείμενα του περιβάλλοντος. Η υλοποίηση της μεθόδου καθορίζει τις διαδικασίες που ενεργοποιούνται στην περίπτωση αυτή.

Κάποια από τα αντικείμενα του περιβάλλοντος του εικονικού κόσμου έχουν υλοποιηθεί με τέτοιον τρόπο ώστε να αλληλεπιδρούν όταν έρχονται σε επαφή με τον

επισκέπτη. Τα αντικείμενα αυτά χρησιμοποιούν τα απαραίτητα datablock που υλοποιούν τις μεθόδους, οι οποίες καθορίζουν την συμπεριφορά των αντικειμένων κατά την αλληλεπίδραση τους με άλλα αντικείμενα. Συγκεκριμένα για τον καθορισμό των δραστηριοτήτων που εκτελούνται κατά την σύγκρουση του αντικειμένου με τον επισκέπτη υλοποιείται η μέθοδος onCollision. Τα datablocks που χρησιμοποιούνται από τα interactive αντικείμενα και οι μέθοδοι που αυτά υλοποιούν βρίσκονται αποθηκευμένα στο αρχείο Museum\server\items.

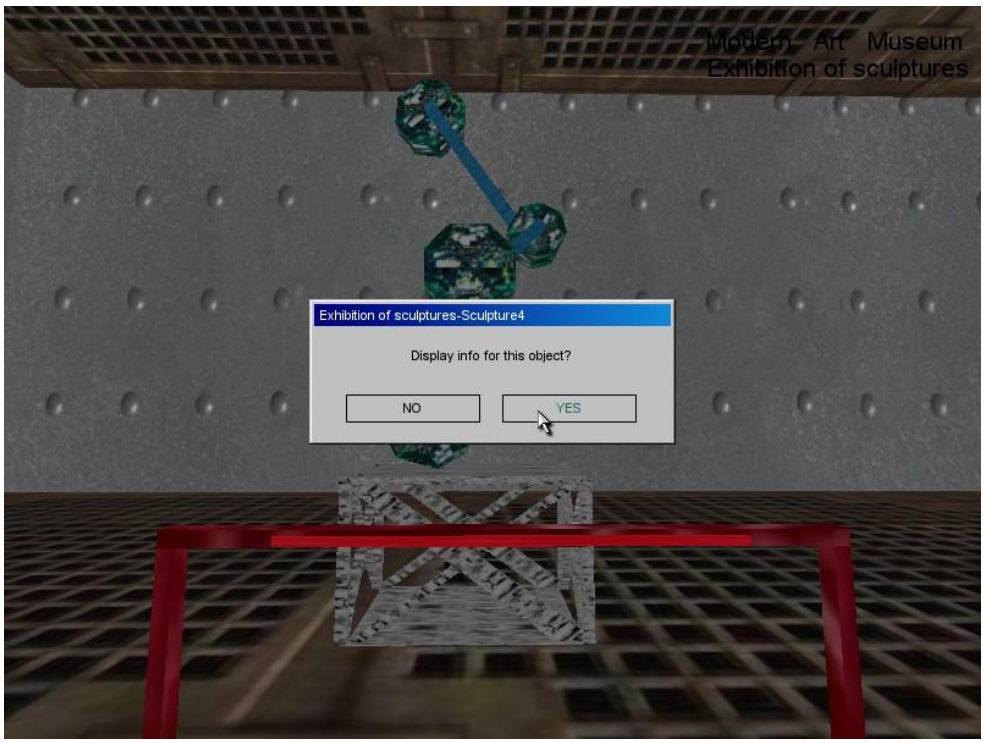
Στον εικονικό κόσμο του μουσείου ως interactive αντικείμενα υλοποιήθηκαν οι προστατευτικές μπάρες μπροστά από τα εκθέματα του μουσείου και το γραφείο πληροφοριών. Έτσι όταν ο επισκέπτης του μουσείου ακουμπάει την προστατευτική μπάρα που βρίσκεται μπροστά από ένα γλυπτό ή πίνακα ζωγραφικής του μουσείου εμφανίζεται παράθυρο διαλόγου που ρωτάει τον χρήστη αν επιθυμεί την εμφάνιση πληροφοριών για το συγκεκριμένο έκθεμα. Αν ο χρήστης το επιθυμεί εμφανίζονται οι πληροφορίες, ενώ αν επιθυμεί περαιτέρω ενημέρωση του δίνεται η δυνατότητα να καλέσει τον ξεναγό του μουσείου μέσω νέου παραθύρου διαλόγου που εμφανίζεται. Σε αυτή την περίπτωση εμφανίζεται ο ξεναγός ο οποίος πλησιάζει το συγκεκριμένο έκθεμα και δίνει στον επισκέπτη περισσότερες πληροφορίες για αυτό. Αντίστοιχα όταν ο επισκέπτης πλησιάζει και ακουμπάει το γραφείο πληροφοριών εμφανίζεται παράθυρο διαλόγου που ρωτάει τον χρήστη αν επιθυμεί να έχει ενημέρωση σχετικά με τις εκθέσεις του μουσείου. Σε περίπτωση που επιθυμεί κάτι τέτοιο εμφανίζεται νέο παράθυρο που του παρέχει τις απαραίτητες πληροφορίες.

Συγκεκριμένα για καθένα από τα αντικείμενα που υλοποιούν τις προστατευτικές μπάρες των εκθεμάτων του μουσείου κατασκευάζεται ένα αντικείμενο της κλάσης StaticShape, στο οποίο αντιστοιχίζεται ένα datablock με όνομα BaseXXItem όπου XX ο αριθμός που αντιστοιχεί στο έκθεμα αυτό. Καθένα από αυτά τα datablocks υλοποιεί την μέθοδο onCollision που καλείται κάθε φορά που το αντικείμενο έρχεται σε επαφή με κάποιο άλλο αντικείμενο. Στην περίπτωση αυτή ελέγχεται αν το αντικείμενο που ήρθε σε επαφή με την προστατευτική μπάρα είναι ο χαρακτήρας του επισκέπτη. Αν πράγματι ο χαρακτήρας αντιστοιχεί στον επισκέπτη δίνεται εντολή στον client να εκτελέσει την συνάρτηση ShowObjectInfo με παράμετρο XX η οποία αναλαμβάνει την εμφάνιση του κατάλληλου παράθυρου διαλόγου προς τον χρήστη. Η εντολή για την εκτέλεση της ShowObjectInfo δίνεται στον client με κλήση της direct messaging συνάρτησης commandToClient. Στην πλευρά του client υλοποιείται η αντίστοιχη συνάρτηση clientCmdShowObjectInfo που δέχεται σαν όρισμα τον αριθμό του εκθέματος και δίνει την δυνατότητα στον χρήστη να επιλέξει την εμφάνιση ή όχι πληροφοριών σχετικά με το έκθεμα μέσω ενός παραθύρου διαλόγου MessageBoxYesNo με μήνυμα κειμένου "Display info for this object? " και με επικεφαλίδα "Exhibition of sculptures-SculptureX" στην περίπτωση που το έκθεμα είναι κάποιο από τα γλυπτά ή "Exhibition of paintings-PaintingX" στην περίπτωση που το έκθεμα είναι κάποιος από τους πίνακες ζωγραφικής.

Στην συνέχεια με κλήση της displayInfo εμφάνίζονται οι σχετικές με το έκθεμα πληροφορίες εφόσον το επιθυμεί ο χρήστης. Συγκεκριμένα εμφανίζεται παράθυρο διαλόγου τύπου MessageBoxYesNo με επικεφαλίδα "Exhibit's Information " και κείμενο που περιγράφει το είδος του εκθέματος, το όνομά του και το όνομα του καλλιτέχνη. Το παράθυρο αυτό πέραν της εμφάνισης των πληροφοριών δίνει στον χρήστη την δυνατότητα να επιλέξει αν επιθυμεί την περαιτέρω ενημέρωση του από τον ξεναγό. Αν επιλέξει την επιλογή Yes καλείται η συνάρτηση GuideInfo που αναλαμβάνει την εμφάνιση του ξεναγού και των επιπλέον πληροφοριών. Συγκεκριμένα κατασκευάζεται ένα αντικείμενο της κλάσης AIManager και καλείται

η συνάρτηση think του αντικειμένου αυτού που αναλαμβάνει την κατασκευή του χαρακτήρα του ξεναγού και την κίνησή του προς το κατάλληλο σημείο.

Εικόνα 6.9 Παράθυρο διαλόγου που εμφανίζεται κατά την σύγκρουση του επισκέπτη με την προστατευτική μπάρα ενός πίνακά ζωγραφικής.



Εικόνα 6.10 Παράθυρο διαλόγου που εμφανίζεται κατά την σύγκρουση του επισκέπτη με την προστατευτική μπάρα ενός γλυπτού.

Το αντικείμενο που υλοποιεί την μπάρα μπροστά στο γραφείο πληροφοριών χρησιμοποιεί το datablock με όνομα InfoDeskItem. Η μέθοδος onCollision που καθορίζει την συμπεριφορά του αντικειμένου αυτού κατά την σύγκρουση του με άλλα αντικείμενα, ελέγχει αρχικά αν το αντικείμενο που συγκρούστηκε με την μπάρα είναι ο χαρακτήρας του επισκέπτη. Σε αυτή την περίπτωση δίνεται εντολή στον client να εκτελέσει την συνάρτηση ShowDeskInfo. Το μήνυμα στον client στέλνεται μέσω της commandToClient. Όταν ο client λάβει το μήνυμα εκτελεί την αντίστοιχη συνάρτηση clientCmdShowDeskInfo η οποία εμφανίζει στην οθόνη παράθυρο διαλόγου MessageBoxYesNo με επικεφαλίδα "Information Desk!" και μήνυμα κειμένου "Welcome to the Information desk! \n Would you like some info?". Έτσι δίνεται η δυνατότητα στον χρήστη να επιλέξει εμφάνιση ή όχι πληροφοριών που αφορούν στις εκθέσεις του μουσείου. Σε περίπτωση που ο χρήστης επιθυμεί την εμφάνιση των πληροφοριών καλείται η συνάρτηση displayDeskInfo που εμφανίζει παράθυρο MessageBoxOK με επικεφαλίδα "General Information" και μήνυμα κειμένου "There are two exhibitions in the museum:\n 1. Exhibition of sculptures \n 2. Exhibition of paintings".

Εικόνα 6.11 Παράθυρο διαλόγου που εμφανίζεται κατά την σύγκρουση του επισκέπτη με το γραφείο πληροφοριών.

Στο παράρτημα παρατίθεται ο κώδικας του αρχείου Museum\server\items (παράγραφος 8.1.2) που περιγράφει τα datablocks που χρησιμοποιούν τα interactive αντικείμενα, οι μέθοδοι onCollision των αντικειμένων αυτών, καθώς και ο κώδικας του αρχείου Museum\client\clientCommands(παράγραφος 8.2.6) που περιγράφει τις συναρτήσεις που ενεργοποιούνται στην πλευρά του client κατά την σύγκρουση του επισκέπτη με κάποιο από τα interactive αντικείμενα

7. ΕΠΙΚΟΙΝΩΝΙΑ ΧΡΗΣΤΗ-ΕΦΑΡΜΟΓΗΣ ΜΕΣΩ GRAPHICAL USER INTERFACES (GUIs)

Ένα μεγάλο πλήθος εφαρμογών multimedia απευθύνονται σε ένα ευρύ κοινό που περιλαμβάνει χρήστες, οι οποίοι δεν έχουν εξειδικευμένες γνώσεις σχετικά με το περιβάλλον των εφαρμογών, τον τρόπο λειτουργίας τους και τις δυνατότητες τους. Επομένως, για να διευκολύνονται οι χρήστες της εφαρμογής, αλλά και για να γίνεται η εφαρμογή προσιτή σε περισσότερους χρήστες είναι απαραίτητη η κατασκευή κατάλληλων διαμεσολαβήσεων (interfaces) για την επικοινωνία χρήστη-εφαρμογής. Τα interfaces πρέπει να είναι απλά και να καθοδηγούν με εύκολο και κατανοητό τρόπο τον χρήστη. Επίσης οφείλουν να παρουσιάζουν στον χρήστη όλες τις εναλλακτικές δυνατότητες της εφαρμογής και να τον διευκολύνουν να επιλέξει την καταλληλότερη για αυτόν.

Έτσι, εξίσου σημαντικό με το περιβάλλον της επίσκεψης στο μουσείο είναι και το περιβάλλον διαμεσολάβησης του χρήστη με την εφαρμογή. Η εφαρμογή πρέπει να παρέχει στον χρήστη την δυνατότητα να εισάγει τις εντολές που επιθυμεί να εκτελεστούν, να επιλέγει την ροή εκτέλεσης της εφαρμογής σε κάποια συγκεκριμένα σημεία αυτής, αλλά και να του παρέχονται πληροφορίες όταν αυτό είναι απαραίτητο.

Η δυνατότητα εισαγωγής των εντολών που επιθυμεί ο χρήστης να εκτελεστούν παρέχεται μέσω της κονσόλας, όμως αυτός ο τρόπος ενδείκνυται μόνο για άτομα που έχουν γνώση της εφαρμογής και του περιβάλλοντος και δεν είναι καθόλου φιλική προς τον χρήστη. Επίσης μέσω της κονσόλας δεν είναι δυνατή η παροχή πληροφοριών προς τον χρήστη ούτε η δυνατότητα επιλογής της ροής εκτέλεσης σε συγκεκριμένα σημεία της εφαρμογής.

Για τους παραπάνω λόγους, πέραν της κονσόλας η επικοινωνία χρήστη εφαρμογής υλοποιείται μέσω γραφικών διαμεσολαβήσεων χρήστη (Graphical User Interfaces) που θα αναφέρονται στη συνέχεια με τα αρχικά GUIs. Τα GUIs παρέχουν έναν φιλικό προς τον χρήστη τρόπο διάδρασης με το πρόγραμμα που δεν προϋποθέτει ειδικές γνώσεις από αυτόν. Επίσης συμβάλλουν στη βελτίωση του αισθητικού αποτελέσματος της εφαρμογής.

Τα γενικά χαρακτηριστικά της μορφής των συστατικών στοιχείων ενός GUI, όπως το χρώμα, το backround, ο αριθμός επιλογών και η γενική μορφή των παραθύρων καθορίζονται στον common κώδικα της εφαρμογής και είναι κοινά για όλα τα στιγμιότυπα που ανήκουν στην ίδια κλάση. Τα υπόλοιπα χαρακτηριστικά τους καθορίζονται από τις τιμές που αποδίδονται στις παραμέτρους τους κατά την κατασκευή τους . Οι τιμές αυτές ενδέχεται να τροποποιηθούν και μετά την κατασκευή ενός αντικειμένου κατά την διάρκεια εκτέλεσης της εφαρμογής.

Ο κώδικας που υλοποιεί το interface του κύριου μενού της εφαρμογής βρίσκεται αποθηκευμένος στο αρχείο mainMenuGui. Το αρχείο clientGui περιλαμβάνει το interface που αναλαμβάνει την επικοινωνία χρήστη- εφαρμογής κατά την πλοήγηση του επισκέπτη στον εικονικό κόσμο. Στο αρχείο loadingGui περιέχεται ο κώδικας που υλοποιεί το interface που εμφανίζεται στην οθόνη κατά την διαδικασία φόρτωσης της εφαρμογής. Τέλος το παράθυρο διαλόγου που εμφανίζεται κατά την επιλογή options του κύριου μενού υλοποιείται με την βοήθεια του κώδικα που περιέχεται στο αρχείο optionsDlg. Τα τέσσερα αρχεία περιλαμβάνονται στον φάκελο Museum\client\ui.

Επιπλέον στοιχεία GUI, όπως παράθυρα διαλόγου και ενημερωτικά μηνύματα που εμφανίζονται κατά την εκτέλεση της εφαρμογής και δεν περιλαμβάνονται στα

παραπάνω αρχεία, υλοποιούνται από τον κώδικα των αρχείων Museum\server\triggers, Museum\client\clientCommands και Museum\client\defaultBind

Τα τέσσερα interfaces που προαναφέρθηκαν και περιλαμβάνονται στα αρχεία mainMenuGui, clientGui, loadingGui και optionsDlg του φακέλου Museum\client\ui παρουσιάζονται αναλυτικά παρακάτω:

7.1 Το interface που υλοποιεί το κύριο μενού της εφαρμογής.

Το αρχείο mainMenuGui περιέχει τον κώδικα που υλοποιεί το GUI του αρχικού μενού της εφαρμογής. Το μενού αυτό, όπως φαίνεται στην εικόνα 7.1, δίνει την δυνατότητα στον χρήστη να επιλέξει μία από τις παρακάτω εναλλακτικές επιλογές :

Visit the museum: επίσκεψη στο μουσείο. Με την επιλογή αυτή ξεκινάει η πλοήγηση στον εικονικό κόσμο της εφαρμογής.

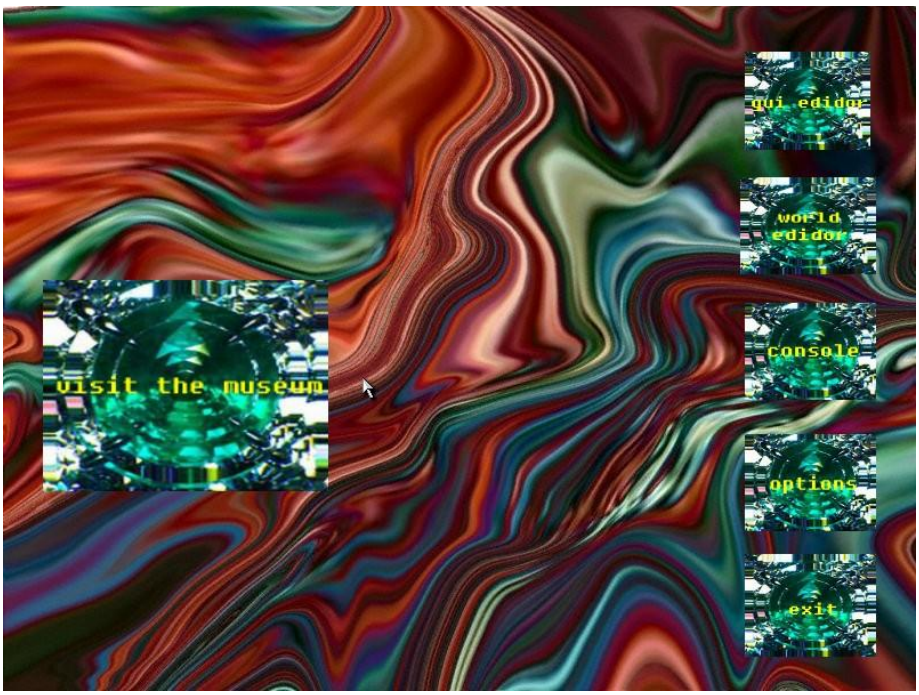
Gui editor άνοιγμα του gui editor του torque. Ανοίγει τον gui editor του Torque game engine προκειμένου να επεξεργαστεί το τρέχον GUI

World editor: άνοιγμα του world editor. Ανοίγει τον world editor του Torque game engine για επεξεργασία του περιβάλλοντος του εικονικού κόσμου και των αντικειμένων που τον απαρτίζουν.

Console: άνοιγμα της κονσόλας. Εμφανίζει το interface της κονσόλας στο οποίο ο χρήστης μπορεί να εισάγει απευθείας εντολές και να πληροφορείται για την πορεία εκτέλεσής τους και τα αποτελέσματα τους.

Options: ρυθμίσεις γραφικών . Εμφανίζει παράθυρο διαλόγου το οποίο δίνεται η δυνατότητα στον χρήστη να κάνει επιλογές σχετικά με τον τρόπο παρουσίασης της εφαρμογής στην οθόνη του, όπως ανάλυση εικόνας, μέγεθος παραθύρου και τύπος γραφικών.

Exit: έξοδος από την εφαρμογή. Η εφαρμογή τερματίζει και κλείνει το αντίστοιχο παράθυρο.



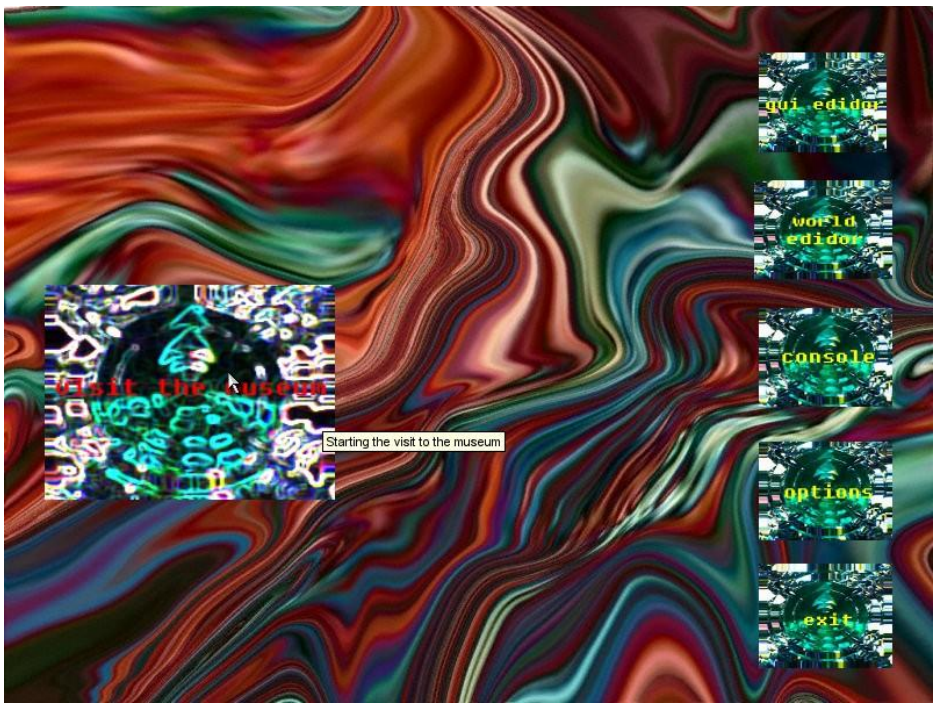
Εικόνα 7.1 Το interface του αρχικού μενού της εφαρμογής.

Το GUI του κύριου μενού της εφαρμογής είναι ένα στιγμιότυπο της κλάσης GuiChunkedBitmapCtrl με όνομα MainMenuGui. Το αντικείμενο αυτό χρησιμεύει ως υπόβαθρο που περιλαμβάνει τα επιμέρους συστατικά στοιχεία του GUIs. Το MainMenuGui ευθυγραμμίζεται με τα άκρα του παραθύρου της εφαρμογής και ρυθμίζεται το μέγεθός του ώστε να το καταλαμβάνει ολόκληρο. Το Gui αυτό καλύπτεται από την εικόνα που βρίσκεται αποθηκευμένη στο αρχείο Museum\client\ui\colours

Το αντικείμενο MainMenuGui περιλαμβάνει 6 αντικείμενα που ανήκουν επίσης στη κλάση GuiChunkedBitmapCtrl. Κατά την δημιουργία τους ορίζονται μερικά από τα χαρακτηριστικά τους όπως το μέγεθός τους, η θέση τους στο κεντρικό παράθυρο καθώς και ο τρόπος που αναπροσαρμόζουν την θέση τους σε τυχόν αλλαγές του μεγέθους του παραθύρου.

Καθένα από τα αντικείμενα αυτά με την σειρά του χρησιμεύει σαν παράθυρο που περιέχει ένα κουμπί τύπου GuiBitmapButtonCtrl. Τα στιγμιότυπα της κλάσης GuiBitmapButtonCtrl είναι κουμπιά που αντιδρούν στο κλικ του ποντικιού όπως υποδηλώνει η τιμή "PushButton" της παραμέτρου buttonType και εκτελούν συγκεκριμένες εντολές. Διαφέρουν από τα κοινά κουμπιά που αποτελούν στιγμιότυπα της κλάσης GuiButtonCtrl μόνο οπτικά και όχι ως προς το αποτέλεσμα που παράγουν. Τα της κλάσης GuiBitmapButtonCtrl έχουν την μορφή κάποιου εικονιδίου που ορίζεται από την παράμετρό τους bitmap και όχι την μορφή ορθογωνίου παραλληλογράμμου. Υπάρχει η δυνατότητα να αλλάζει το εικονίδιο του κουμπιού όταν ο κέρσορας του ποντικιού βρίσκεται πάνω στο κουμπί. Αυτό επιτυγχάνεται με αποθήκευση δύο εικόνων με το όνομα του αρχείου της παραμέτρου bitmap και με καταλήξεις _n(αντιστοιχεί στην αρχική μορφή του εικονιδίου) και _h (αντιστοιχεί στην μορφή του εικονιδίου όταν ο κέρσορας του ποντικιού βρίσκεται πάνω στο κουμπί). Το πάτημα του κάθε κουμπιού προκαλεί την εκτέλεση μιας εντολής που ορίζεται από το πεδίο Command του αντικειμένου. Όταν ο κέρσορας του ποντικιού βρίσκεται πάνω στο κουμπί εμφανίζεται ένα μήνυμα που πληροφορεί τον

χρήστη για την ενέργεια που εκτελείται κατά το πάτημα του κουμπιού. Το μήνυμα αυτό αποθηκεύεται στην παράμετρο tooltip του αντικειμένου. Η μορφή του μηνύματος ακολουθεί τα χαρακτηριστικά του προφίλ GuiToolTipProfile που ορίζεται ως τιμή της παραμέτρου tooltipprofile.



Εικόνα 7. 2 Το interface του αρχικού μενού της εφαρμογής, με τον κέρσορα να επιλέγει μία από τις δυνατές επιλογές.

Η εντολή που εκτελείται κατά το πάτημα του καθενός από τα 6 κουμπιά, το εικονίδιο του καθενός και το μήνυμα που εμφανίζεται όταν ο κέρσορας του ποντικιού βρίσκεται πάνω του φαίνονται στον πίνακα 7.1.

Α/Α	Κουμπί (button)	Εντολή (command)	Εικονίδιο (bitmap)	Μήνυμα (tooltip)
1	Visit the museum	loadMyMission()	Start	Starting the visit to the museum
2	Gui editor	GuiEdit()	Guieditor	Open the GUI Editor, which helps you create graphical user interfaces for your games
3	World editor	ToggleEditor(1)	Worldeditor	Open the World Editor, which provides tools for creating and editing your game worlds
4	Console	ToggleConsole(1)	Consol	View the scripting Console, which allows you to enter TorqueScript commands on the fly
5	Options	Canvas.pushDialog(optionsDlg)	Option	Adjust basic video, and other options
6	Exit	quit()	Exits	Close down the engine

Πίνακας 7.1 Οι επιλογές του main menu.

Ο κώδικάς του αρχείου Museum\client\ui\mainMenuGui που υλοποιεί το παραπάνω GUI παρατίθεται στη παράγραφο 8.2.11 του παραρτήματος.

7. 2 Το interface της πλοήγησης στον εικονικό κόσμο.

Το αρχείο clientGui περιέχει τον κώδικα που υλοποιεί το interface που εμφανίζεται κατά την πλοήγηση του επισκέπτη στον εικονικό κόσμο. Το interface αυτό αναλαμβάνει την εμφάνιση μηνυμάτων που παρέχουν πληροφορίες στον χρήστη και διευκολύνουν την επικοινωνία του με την εφαρμογή. Τα περισσότερα από τα μηνύματα αυτά αρχικά είναι αόρατα στον χρήστη, δηλαδή η τιμή της παραμέτρου τους visible είναι μηδέν, και εμφανίζονται όταν αυτό κρίνεται απαραίτητο για την πληροφόρηση του χρήστη.

Στο αρχείο clientGui αρχικά κατασκευάζεται ένα καινούριο αντικείμενο με όνομα playGui το οποίο αποτελεί στιγμιότυπο της κλάσης GameTSCtrl. Σκοπός του αντικειμένου αυτού είναι να αποτελέσει το υπόβαθρο στο οποίο στη συνέχεια θα

προστεθούν τα υπόλοιπα συστατικά στοιχεία του GUI. Τα στοιχεία αυτά και η χρήση τους στην εφαρμογή παρουσιάζονται αναλυτικά παρακάτω.

Το αντικείμενο MuseumTag αποτελεί στιγμιότυπο της κλάσης GuiTextCtrl. Τα αντικείμενα της κλάσης GuiTextCtrl εμφανίζουν στην οθόνη ένα μήνυμα κειμένου το οποίο βρίσκεται αποθηκευμένο στην παράμετρο με όνομα text. Τα χαρακτηριστικά της γραμματοσειράς που χρησιμοποιείται καθορίζονται από το προφίλ του αντικειμένου, δηλαδή την τιμή της παραμέτρου Profile. Στην συγκεκριμένη περίπτωση το προφίλ που χρησιμοποιείται είναι το GuiMediumTextProfile και το μήνυμα που εμφανίζεται είναι Modern Art Museum. Το αντικείμενο αυτό είναι το μόνο συστατικό του clientGui που παραμένει ορατό καθ' όλη την διάρκεια της εφαρμογής και πληροφορεί τον χρήστη ότι ο εικονικός κόσμος τον οποίο επισκέπτεται αποτελεί το περιβάλλον ενός μουσείου μοντέρνας τέχνης. Το μήνυμα εμφανίζεται στην πάνω δεξιά γωνία της οθόνης, όπως καθορίζεται από τις τιμές των παραμέτρων που αφορούν στη θέση και έκταση του μηνύματος. Όταν μεταβάλλεται το μέγεθος και η θέση του παραθύρου που περιέχει το GUI το αντικείμενο MuseumTag μεταβάλλεται κατά ανάλογο τρόπο, όπως ορίζει η τιμή relative των παραμέτρων HorizSizing και VertSizing.

Το αντικείμενο helpMessage αποτελεί στιγμιότυπο της κλάσης GuiTextCtrl. Το προφίλ που χρησιμοποιείται είναι το GuiTextProfile και το μήνυμα που αποτελεί τιμή της παραμέτρου text είναι Push H for help. Το αντικείμενο αυτό του clientGui εμφανίζεται κατά την εναρξη της πλοήγησης στον εικονικό κόσμο και παραμένει ορατό μερικά δευτερόλεπτά. Πληροφορεί τον χρήστη ότι με πάτημα του πλήκτρου H εμφανίζεται παράθυρο πληροφοριών. Το μήνυμα εμφανίζεται στην πάνω δεξιά γωνία της οθόνης κάτω από την επιγραφή Modern Art Museum. Το αντικείμενο helpMessage προσαρμόζει το μέγεθος και την θέση του ανάλογα με το μέγεθος και η θέση του παραθύρου που περιέχει το GUI

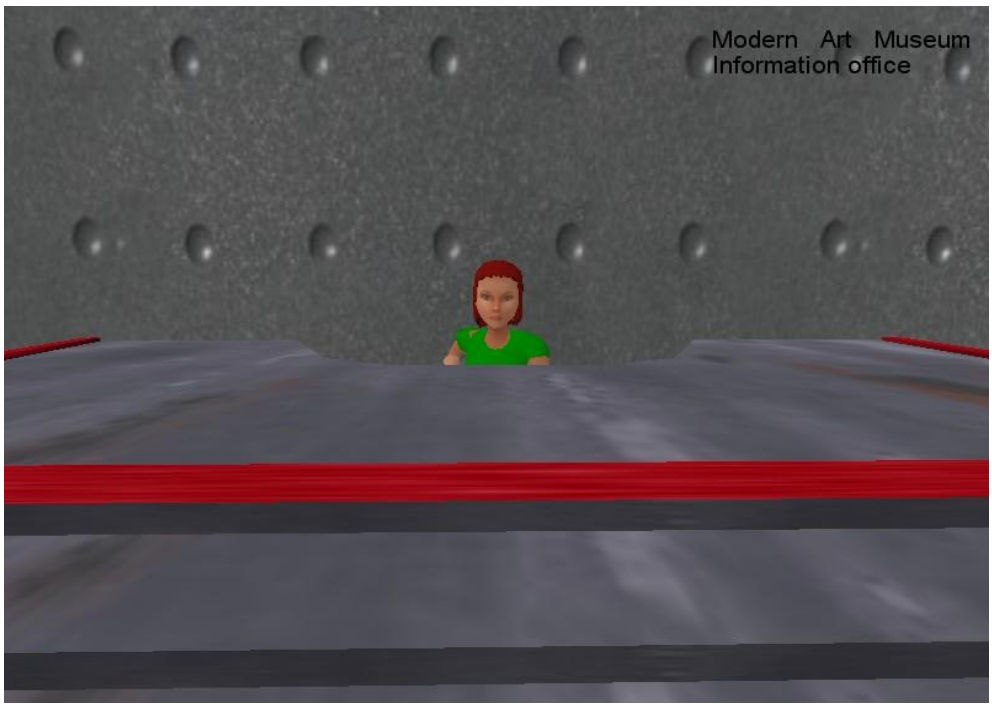
Τα αντικείμενα StartTriggerControler1 και StartTriggerControler2 αποτελούν επίσης στιγμιότυπα της κλάσης GuiTextCtrl, όμως η γραμματοσειρά που χρησιμοποιούν για την εμφάνιση του μηνύματος τους έχει μεγαλύτερο μέγεθος από αυτή του MuseumTag αφού χρησιμοποιούν το προφίλ GuiBigTextProfile. Τα αντικείμενα αυτά εμφανίζουν τα μηνύματα Information Office και Modern Art Museum κατά την έναρξη της εφαρμογής προκειμένου να ενημερώσουν τον επισκέπτη σχετικά με τα δύο κτίρια που εμφανίζονται μπροστά του. Τα δύο μηνύματα εμφανίζονται ακριβώς πάνω από τα αντίστοιχα κτίρια για λίγα δευτερόλεπτα και στη συνέχεια εξαφανίζονται.. Αυτό επιτυγχάνεται με αλλαγή της τιμής της παραμέτρου visible σε 1 κατά την εισαγωγή του επισκέπτη στο trigger starttrigger η οποία καθιστά τα μηνύματα ορατά. Μετά από λίγα δευτερόλεπτα η τιμή της παραμέτρου αλλαζει σε 0 με αποτέλεσμα την εξαφάνιση των μηνυμάτων από την οθόνη. Το μήνυμα εμφανίζεται κατά την έναρξη της εφαρμογής, αφού το starttrigger βρίσκεται γύρω από το σημείο αρχικής εμφάνισης του επισκέπτη στον εικονικό κόσμο.

Το αντικειμενο PlaceTriggerControler της κλάσης GuiTextCtrl έχει σκοπό να ενημερώνει τον επισκέπτη σε ποιο χώρο βρίσκεται κάθε φορά. Έχει προφιλ GuiMediumTextProfile και εμφανίζει κάτω από την επιγραφή Modern Art Museum στην πάνω δεξιά γωνία της οθόνης ένα μήνυμα με το όνομα του χώρου. Το μήνυμα αυτό είναι αρχικά αόρατο και γίνεται ορατό κατά την είσοδο του επισκέπτη στο γραφείο πληροφοριών του μουσείου ή σε κάποια από τις εκθέσεις του. Η αλλαγή της τιμής της παραμέτρου visible από 0 σε 1 και η επιλογή του κατάλληλου μηνύματος γίνεται από τον κώδικα της συνάρτησης onEnterTrigger του trigger που αντιστοιχεί στον κάθε χώρο. Έτσι κατά την είσοδο του επισκέπτη στο γραφείο πληροφοριών του μουσείου εμφανίζεται το μήνυμα Information office, κατά την είσοδό του στην

έκθεση γλυπτών εμφανίζεται το μήνυμα Exhibition of sculptures, ενώ κατά την είσοδό του στην έκθεση ζωγραφικής εμφανίζεται το μήνυμα Exhibition of paintings τα οποία παραμένουν ορατά καθ' όλη την διάρκεια παραμονής του στον κάθε χώρο.

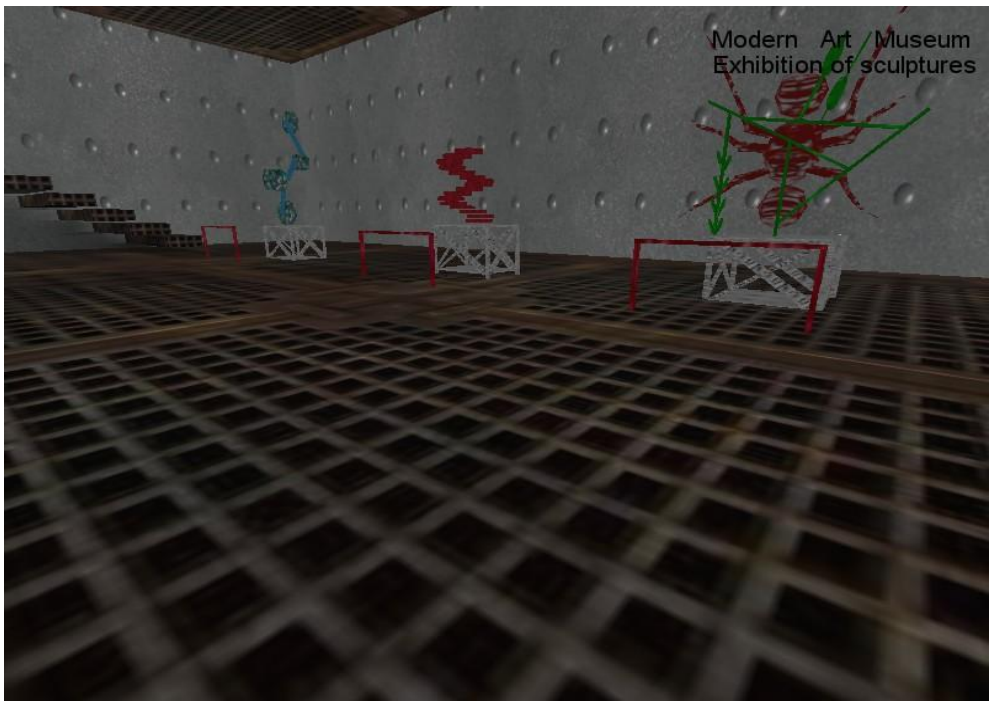
Το αντικείμενο ObjectTriggerControler είναι στιγμιότυπο της κλάσης GuiTextEditCtrl και εμφανίζει το όνομα του καλλιτέχνη και του έργου όταν ο επισκέπτης ζητήσει πληροφορίες σχετικά με κάποιο έκθεμα από τον ξεναγό του μουσείου. Το κείμενο εμφανίζεται μέσα σε ένα άσπρο πλαίσιο και το περιεχόμενο του καθορίζεται από το trigger που αντιστοιχεί σε κάθε έκθεμα, όταν ο χαρακτήρας του ξεναγού εισέρχεται στο χώρο του trigger. Οι υπόλοιπες παράμετροι του αντικειμένου ορίζουν το μέγεθός του, τη θέση του στο παράθυρο καθώς και τον τρόπο που αναπροσαρμόζει την θέση του σε τυχόν αλλαγές του μεγέθους του παραθύρου.

Τα αντικείμενα ArtistInfoTriggerControler και ExhibitInfoTriggerControler ανήκουν επίσης στην κλάση GuiTextEditCtrl και εμφανίζουν επιπλέον πληροφορίες σχετικά με τον καλλιτέχνη και το έργο του. Το κείμενο εμφανίζεται μέσα σε ένα άσπρο πλαίσιο και το περιεχόμενο του καθορίζεται από το trigger που αντιστοιχεί σε κάθε έκθεμα, όταν ο χαρακτήρας του ξεναγού εισέρχεται στο χώρο του trigger, όπως και στην περίπτωση του ObjectTriggerControler

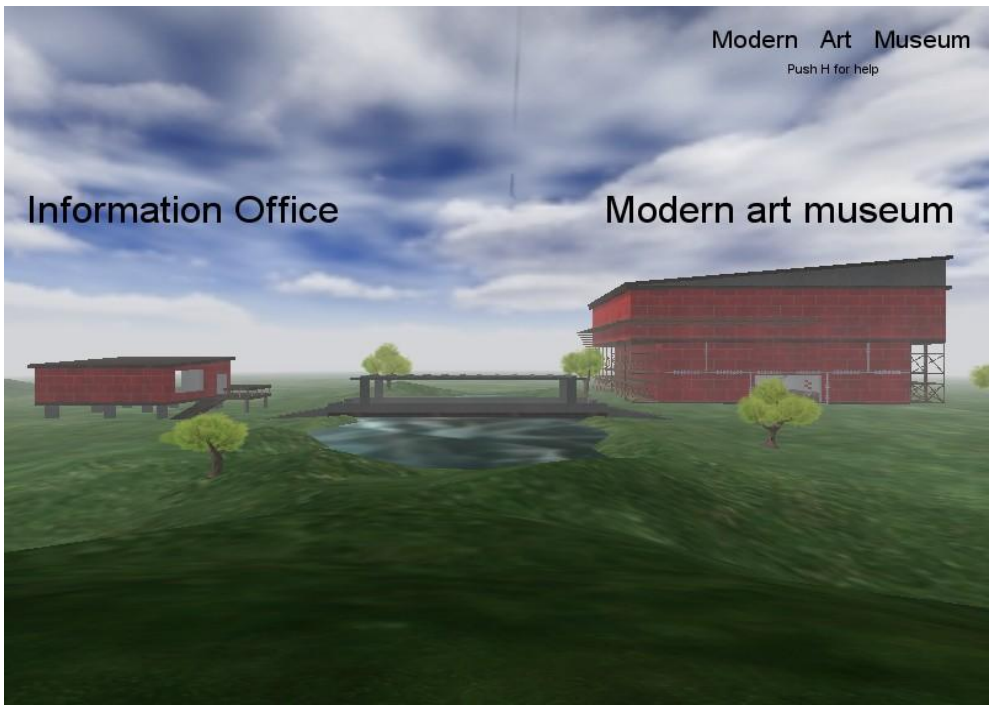


Εικόνα 7. 3 Εμφάνιση του μηνύματος 'Information office', κατά την είσοδο στο γραφείο πληροφοριών.

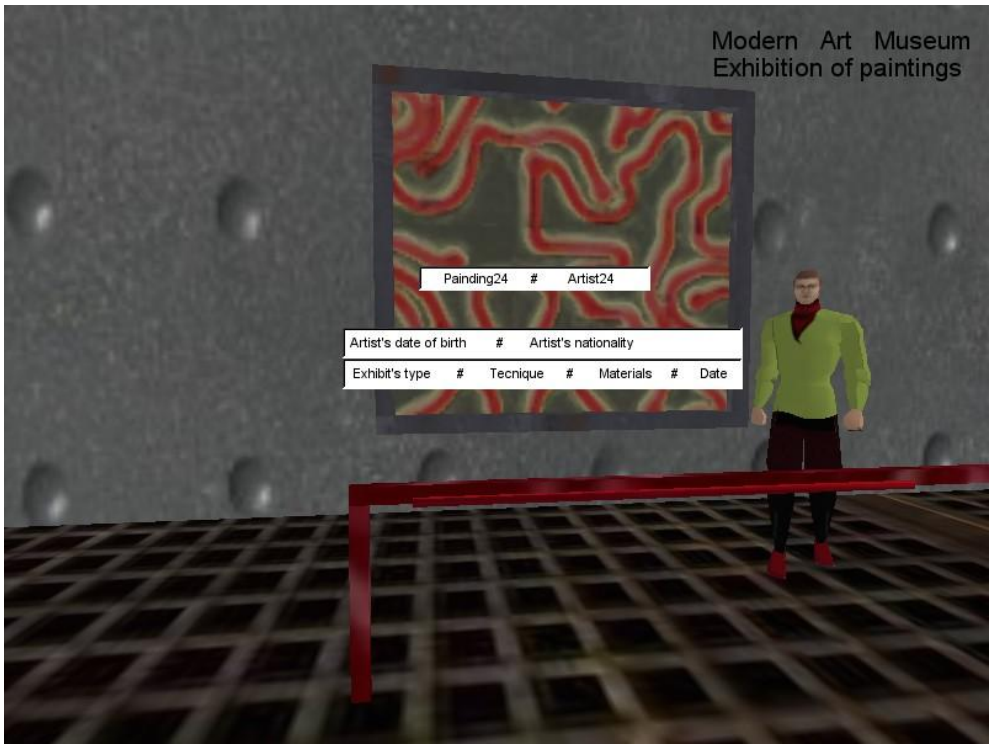
Εικόνα 7. 4 Εμφάνιση του μηνύματος 'Exhibition of paintings', κατά την είσοδο στην έκθεση ζωγραφικής.



Εικόνα 7. 5 Εμφάνιση του μηνύματος 'Exhibition of sculptures', κατά την είσοδο στην έκθεση γλυπτών.



Εικόνα 7. 6 Εμφάνιση πληροφοριακών μηνυμάτων , κατά την έναρξη της εφαρμογής.



Εικόνα 7.7 Πληροφορίες για το έκθεμα και τον καλλιτέχνη από τον ξεναγό του μουσείου.

Ο κώδικάς του αρχείου Museum\client\ui\clientGui που υλοποιεί το παραπάνω GUI παρατίθεται στο παράρτημα του κεφαλαίου 8, στη παράγραφο 8.2.9.

7.3 Το interface που υλοποιεί το παράθυρο φόρτωσης της εφαρμογής.

Το αρχείο loadingGui περιλαμβάνει τον κώδικα που εμφανίζει το παράθυρο ενημέρωσης του χρήστη κατά την διάρκεια φόρτωσης(loading) της εφαρμογής, λίγο πριν την έναρξη της πλοήγησης του επισκέπτη στον εικονικό κόσμο. Σκοπός αυτού του interface είναι να ενημερώνει τον χρήστη για την πορεία του loading καθώς και να του παρέχει την δυνατότητα να το διακόψει, σε περίπτωση που αλλαξει γνώμη, μέσω του πλήκτρου cancel.

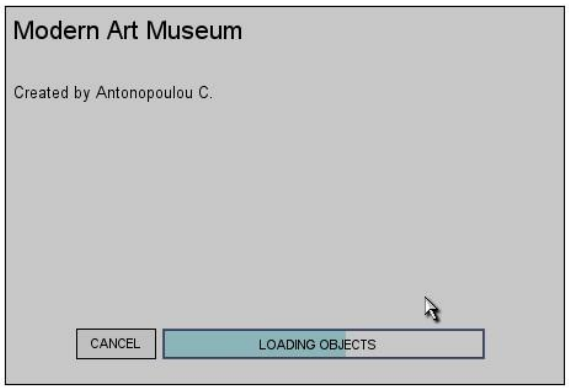
Το βασικό αντικείμενο του Gui που περιλαμβάνει και τα υπόλοιπα ονομάζεται LoadingGui και είναι στιγμιότυπο της κλάσης GuiChunkedBitmapCtrl. Το αντικείμενο αυτό υλοποιεί το παράθυρο που περιέχει τα υπόλοιπα συστατικά στοιχεία του Gui. Τα στοιχεία αυτά περιέχονται σε ένα αντικείμενο της κλάσης GuiControl που αποτελεί συστατικό του LoadingGui και παρουσιάζονται παρακάτω

Το αντικείμενο LOAD_MapName είναι ένα αντικείμενο της κλάσης GuiTextCtrl με προφίλ GuiMediumTextProfile και μήνυμα κειμένου Map Name.

Το αντικείμενο LoadingProgress. είναι στιγμιότυπο της κλάσης GuiProgressCtrl και περιλαμβάνει δύο επιμέρους αντικείμενα. Το ένα εμφανίζει την μπάρα που δείχνει την πορεία του loading και ένα πλήκτρο ακύρωσης του loading αν ο χρήστης το επιθυμεί. Τα αντικείμενα αυτά είναι τα LoadingProgressTxt και cancelButton

Το αντικείμενο LoadingProgressTxt είναι στιγμιότυπο της κλάσης GuiTextCtrl με προφίλ GuiProgressTextProfile και εμφανίζει στο παράθυρο του interface μία μπάρα που παρουσιάζει την πορεία της φόρτωσης τον αρχείων της εφαρμογής και περιέχει το μήνυμα κειμένου LOADING MISSION.

Το αντικείμενο cancelButton. Αποτελεί στιγμιότυπο της κλάσης GuiButtonCtrl και υλοποεί το πλήκτρο που δίνει την δυνατότητα στο χρήστη να διακόψει την φόρτωση της εφαρμογής εφόσον το επιθυμεί. Φέρει το μήνυμα κειμένου CANCEL και χρησιμοποιεί το προφίλ GuiButtonProfile. Η εντολή που εκτελείται κατά το πάτημα του κουμπιού είναι η disconnect(). Στο κουμπί cancelButton αντιστοιχίζεται το πλήκτρο συντόμευσης escape, πάτημα του οποίου επιφέρει τα ίδια αποτελέσματα, δηλαδή διακοπή της φόρτωσης.



Εικόνα 7.8 Το interface του παραθύρου φόρτωσης της εφαρμογής.

Ο κώδικάς του αρχείου Museum\client\ui\loadingGui που υλοποιεί το παραπάνω GUI παρατίθεται στο παράρτημα του κεφαλαίου 8..

7.4 Το interface που υλοποιεί το παράθυρο ρυθμίσεων.

Το αρχείο optionsDlg περιλαμβάνει τον κώδικα που υλοποιεί το παράθυρο διαλόγου που εμφανίζεται όταν ο χρήστης επιλέξει την επιλογή options του main menu. Αυτή η επιλογή του δίνει την δυνατότητα να πραγματοποιήσει ρυθμίσεις σχετικά με την ανάλυση της εικόνας, τα γραφικά και το μέγεθος του παραθύρου. Οι ρυθμίσεις που επιλέγει ο χρήστης αποθηκεύονται σε κατάλληλες μεταβλητές προκειμένου να χρησιμοποιηθούν στην συνέχεια από την εφαρμογή και να της δώσουν την μορφή που επιθυμεί ο χρήστης.

Συγκεκριμένα ο κώδικάς του αρχείου κατασκευάζει ένα αντικείμενο τύπου GuiControl με όνομα optionsDlg. Το βασικό συστατικό του optionsDlg που υλοποιεί το παράθυρο, όπου εμφανίζονται οι δυνατότητες επιλογής είναι στιγμιότυπο της κλάσης GuiWindowCtrl. Στο αντικείμενο αυτό εμπεριέχονται τα συστατικά στοιχεία που περιγράφονται στη συνέχεια.

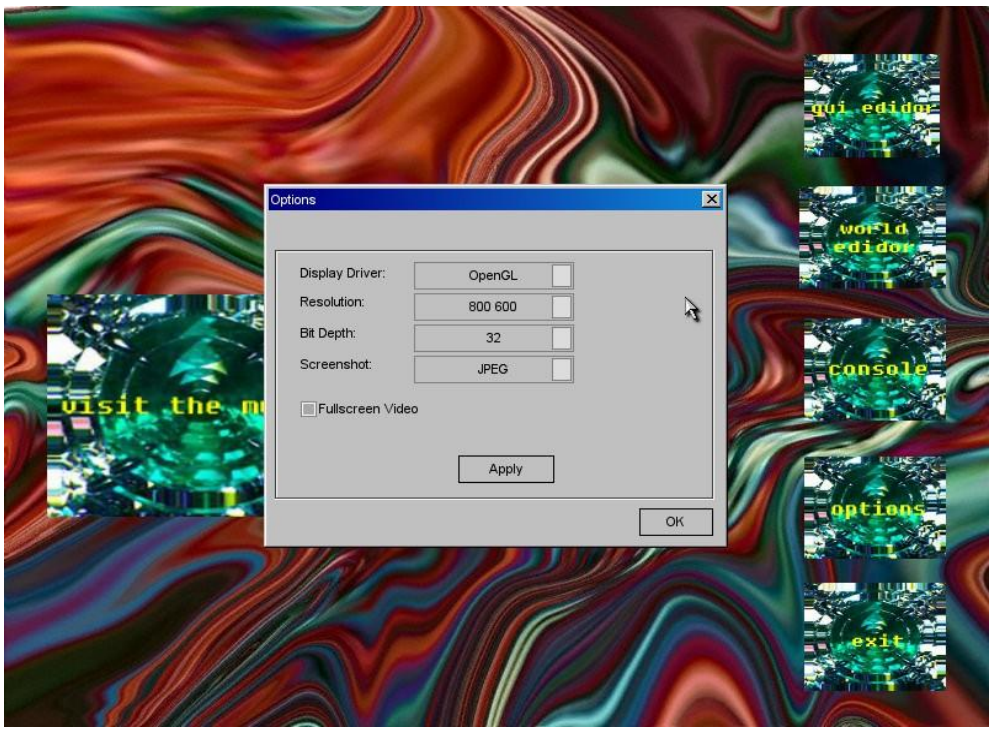
Το αντικείμενο OK_Button.είναι στιγμιότυπο της κλάσης GuiButtonCtrl και ενεργοποιείται με το κλικ του ποντικιού όπως ορίζει η τιμή PushButton της παραμέτρου buttonType. Ενεργοποίηση του κουμπιού έχει σαν αποτέλεσμα την επιστροφή στο κύριο μενού της εφαρμογής με κλήση της μεθόδου Canvas.popDialog(optionsDlg), που αποτελεί τιμή της παραμέτρου Command. Το προφίλ που χρησιμοποιεί το αντικείμενο είναι το GuiButtonProfile και φέρει το μήνυμα κειμένου "OK".

Το panel OptGraphicsPane της κλάσης GuiControl περίέχει τα παρακάτω συστατικά στοιχεία που αποτελούν και τις εναλλακτικές επιλογές του χρήστη.

Το αντικείμενο OptGraphicsFullscreenToggle ανήκει στην κλάση GuiCheckBoxCtrl και υλοποιεί μία επιλογή που ενεργοποιείται με τσεκάρισμα του τετραγωνιδίου που φέρει. Δίπλα απ' το τετραγωνίδιο εμφανίζεται το μήνυμα Fullscreen Video. Τσεκάρισμα του τετραγωνιδίου αντιστοιχεί σε αποθήκευση της τιμής 1 στην μεταβλητή pref::Video::fullScreen. Η τιμή αυτή έχει σαν αποτέλεσμα την αναπροσαρμογή του παραθύρου ώστε να καταλαμβάνει ολόκληρη την οθόνη. Σε αντίθετη περίπτωση η μεταβλητή παίρνει την τιμή 0 και το παράθυρο της εφαρμογής καταλαμβάνει μικρότερες διαστάσεις. Τα χαρακτηρηστικα του αντικειμένου καθορίζονται από το προφίλ GuiCheckBoxProfile, και ο τύπος της επιλογής είναι ToggleButton, όπως υποδηλώνει η τιμή της παραμέτρου buttonType.

Στη συνέχεια κατασκευάζονται 4 αντικείμενα που ανήκουν στην κλάση GuiPopUpMenuCtrl, με ονόματα OptGraphicsDriverMenu, OptGraphicsResolutionMenu, OptGraphicsBPPMenu, OptScreenshotMenu και ισάριθμα αντικείμενα της κλάσης GuiTextCtrl, με ονόματα GraphicsDriverText, GraphicsResolutionText, GraphicsBPPText, ScreenshotText. Τα αντικείμενα τύπου GuiPopUpMenuCtrl υλοποιούν 4 PopUp μενού τα οποία δίνουν την δυνατότητα στον χρήστη να κάνει επιλογές σχετικά με τον Driver γραφικών που θα χρησιμοποιηθεί, την ανάλυση της εικόνας, το bit depth και τον τύπο αρχείου που θα χρησιμοποιηθεί για αποθήκευση των screenshot, αντίστοιχα. Καθένα από τα αντικείμενα τύπου GuiTextCtrl εμφανίζει ένα μήνυμα κειμένου δίπλα από κάθε PopUp μενού με σκοπό να πληροφορήσει τον χρήστη για τις δυνατότητες επιλογής που του παρέχονται. Τα μηνύματα που εμφανίζονται είναι Display Driver, Resolution, Bit Depth και Screenshot κατά αντιστοιχία με τα μενού OptGraphicsDriverMenu, OptGraphicsResolutionMenu, OptGraphicsBPPMenu, OptScreenshotMenu.

Τέλος κατασκευάζεται ένα στιγμιότυπο της κλάσης GuiButtonCtrl με όνομα Apply_Button που εμφανίζει στο παράθυρο επιλογών ένα κουμπί που φέρει το μήνυμα Apply και έχει ως σκοπό την δυνατότητα επικύρωσης των επιλογών του χρήστη. Το κουμπί αυτό ενεργοποιείται με κλίκ του ποντικιού, όπως καθορίζεται από την τιμή PushButton, της παραμέτρου buttonType και χρησιμοποιεί το προφίλ GuiButtonProfile. Ενεργοποίηση του κουμπιού έχει σαν αποτέλεσμα την κλήση της συνάρτησης optionsDlg.applyGraphics() που αποτελεί την τιμή της παραμέτρου Command του αντικειμένου



Εικόνα 7. 9 Το interface του παράθυρου ρυθμίσεων.

Ο κώδικάς του αρχείου Museum\client\ui\optionsDlg που υλοποιεί το παραπάνω GUI παρατίθεται στη παράγραφο 8.2.12 του παραρτήματος.

7.5 Παράθυρα διαλόγου που υλοποιούνται από τον κώδικα άλλων αρχείων.

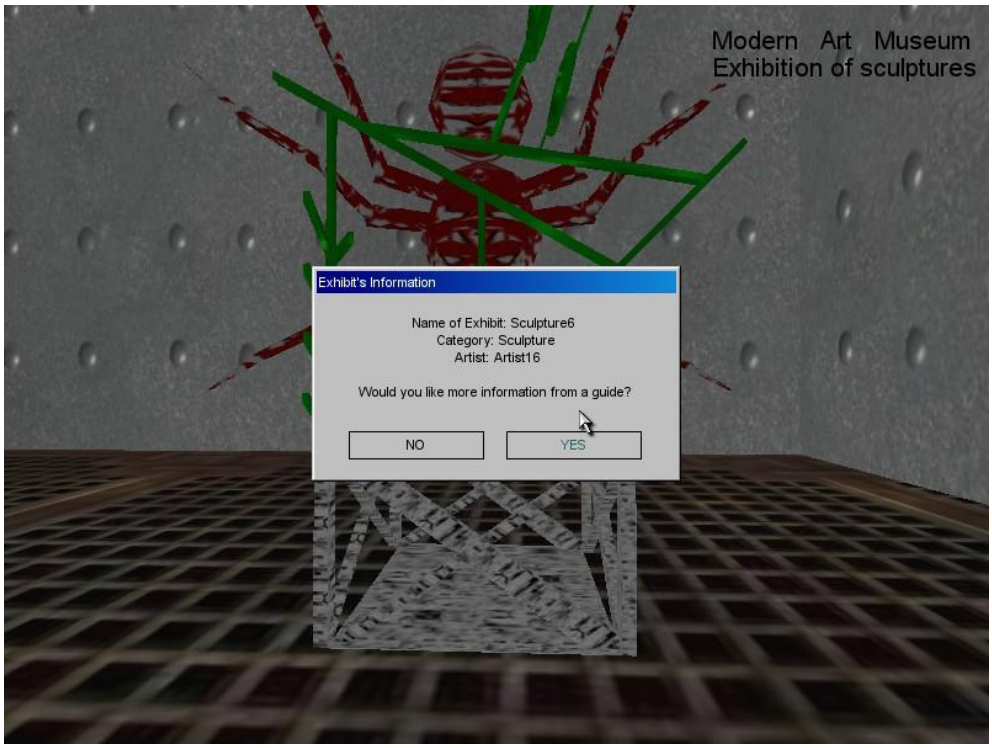
Άλλα αντικείμενα Gui που δεν περιέχονται στα παραπάνω αρχεία αλλά εμφανίζονται κατά την πλοήγηση του επισκέπτη στον εικονικό κόσμο περιλαμβάνονται στον κώδικα των αρχείων Museum\server\triggers, Museum\client\clientCommands και Museum\client\defaultBind. Τα αντικείμενα αυτά εμφανίζονται κατά την είσοδο του επισκέπτη σε χώρους που καταλαμβάνονται από triggers ή κατά την αλληλεπίδραση του με αντικείμενα του περιβάλλοντος. Υλοποιούν παράθυρα διαλόγου που έχουν σκοπό είτε να ενημερώσουν τον επισκέπτη ή να του δώσουν την δυνατότητα να κάνει επιλογές που επηρεάζουν την ροή εκτέλεσης της εφαρμογής.

Τα αντικείμενα αυτά είναι τριών τύπων MessagePopup, MessageBoxOK και MessageBoxYesNo. Τα δύο πρώτα αντικείμενα έχουν χρησιμοποιηθεί για να παρέχουν απλώς κάποια πληροφορία στον χρήστη και διαφέρουν μεταξύ τους στο ότι τα αντικείμενα τύπου MessagePopup εξαφανίζονται λίγα δευτερόλεπτα μετά την εμφάνισή τους χωρίς παρέμβαση του χρήστη, ενώ τα αντικείμενα τύπου MessageBoxOK παραμένουν στην οθόνη για όσο χρονικό διάστημα επιθυμεί ο χρήστης και εξαφανίζονται με πάτημα του κουμπιού OK που φέρουν.

Παράθυρα διαλόγου τύπου MessagePopup έχουν χρησιμοποιηθεί στην εφαρμογή κατά την είσοδο του επισκέπτη στο γραφείο πληροφοριών του μουσείου ή στον χώρο

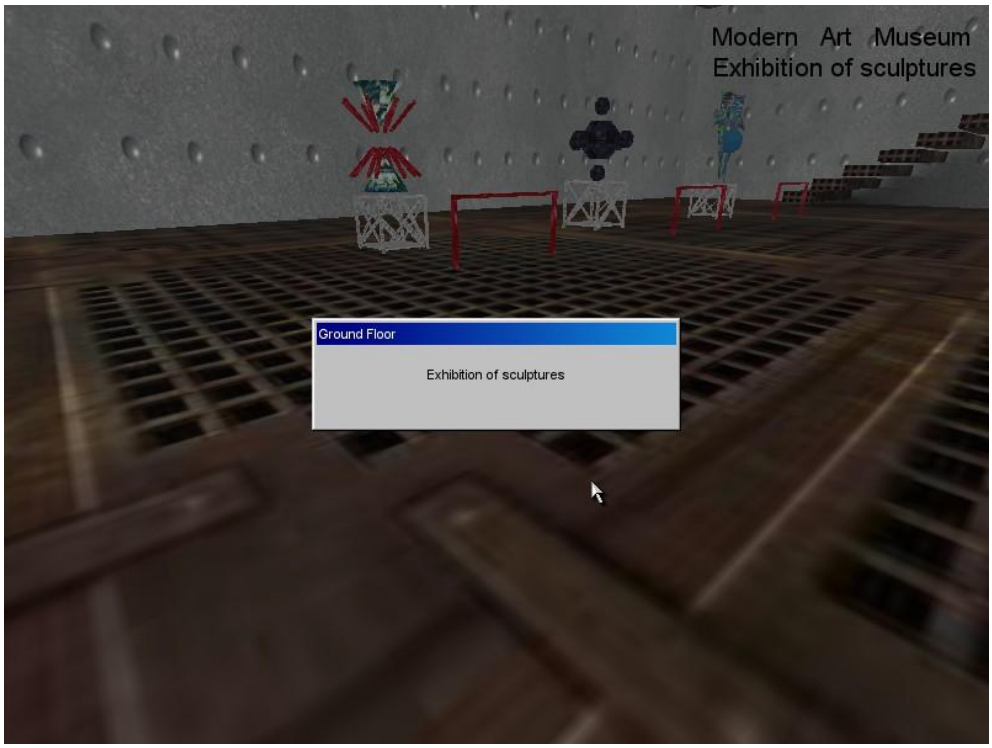
κάποιας από τις δύο εκθέσεις του και έχουν σαν σκοπό την ενημέρωση του για τον χώρο αυτό. Αντικείμενα τύπου MessageBoxOK χρησιμοποιούνται κατά την παρουσίαση πληροφοριών σχετικά με τις εκθέσεις του μουσείου όταν ο χρήστης ζητήσει κάτι τέτοιο από το γραφείο πληροφοριών. Πάτημα του πλήκτρου 'Η' έχει επίσης σαν αποτέλεσμα την εμφάνιση παραθύρου MessageBoxOK που ενημερώνει τον χρήστη σχετικά με την κίνηση του χαρακτήρα του επισκέπτη.

Αντίθετα τα παράθυρα διαλόγου τύπου MessageBoxYesNo δίνουν την δυνατότητα στον χρήστη να παρέμβει στην ροή εκτέλεσης της εφαρμογής κάνοντας κατάλληλες επιλογές. Τέτοιου είδους παράθυρα εμφανίζονται όταν ο επισκέπτης πλησιάζει κάποιο έκθεμα και του ζητείται να αποφασίσει αν επιθυμεί να έχει πληροφόρηση σχετικά με το αντικείμενο αυτό. Στην συνέχεια εμφανίζεται νέο παράθυρο διαλόγου MessageBoxYesNo το οποίο δίνει στον χρήστη την δυνατότητα να επιλέξει αν επιθυμεί την εμφάνιση ή όχι του ξεναγού προκειμένου να αποκτήσει περισσότερη γνώση σχετικά με το συγκεκριμένο έκθεμα. Τέλος όταν ο χρήστης πατήσει το πλήκτρο escape εμφανίζεται παράθυρο διαλόγου MessageBoxYesNo που του δίνει την δυνατότητα να εγκαταλείψει την επίσκεψη στον εικονικό κόσμο και να επιστρέψει στο κύριο μενού της εφαρμογής ή να συνεχίσει την πλοήγησή του.



Εικόνα 7.10 Παράθυρο διαλόγου τύπου MessageBoxYesNo

Εικόνα 7.11 Παράθυρο διαλόγου τύπου MessageBoxOK



Εικόνα 7. 12 Παράθυρο τύπου MessageBoxPopup

Ο κώδικας των συναρτήσεων clientCmdShowDeskInfo, displayDeskInfo,clientCmdShowObjectInfo και displayInfo που υλοποιούν παράθυρα διαλόγου τύπου MessageBoxYesNo και MessageBoxOK βρίσκονται στο αρχείο

Museum\client\clientCommands και παρατίθεται στο παράρτημα, στην παράγραφο 8.2.6.

Στο παράρτημα του κεφαλαίου 8 παρουσιάζεται, επίσης ο κώδικας των μεθόδων onEnterTrigger των αντικειμένων τύπου ExpoOneTrigger, ExpoTwoTrigger και InfoHouseTrigger του αρχείου Museum\server\triggers(παράγραφος 8.1.3), που υλοποιούν παράθυρα διαλόγου τύπου MessagePopup.

Τέλος, στην παράγραφο 8.2.6, παρατίθεται ο κώδικας των συναρτήσεων escapeFromGame και displayHelpMessage του αρχείου Museum\client\defaultBind. Η συνάρτηση escapeFromGame εμφανίζει παράθυρο διαλόγου κατά το πάτημα του πλήκτρου escape, ενώ η συνάρτηση displayHelpMessage εμφανίζει παράθυρο με πληροφορίες κατά το πάτημα του πλήκτρου ‘H’.

8. ΠΑΡΑΡΤΗΜΑ: Ο ΚΩΔΙΚΑΣ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Στις παραγράφους που ακολουθούν παρουσιάζεται ο κώδικας του τμήματος εξηπυρετητή, του τμήματος πελάτη και του τμήματος δεδομένων της εφαρμογής.

8.1 Εξυπηρετητής (server).

8.1.1 Museum\server\museumCode.cs

```
//-----------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C
// Museum\server\museumCode.cs
//-----------------------------------------------------------------------

//-----------------------------------------------------------------------
//  Server and mission initialization
//-----------------------------------------------------------------------

function onServerCreated()
{
   // This function is called when a server is constructed.

   // Master server information for multiplayer games
   $Server::GameType = "Torque TTB";
   $Server::MissionType = "None";

   // Load up all datablocks, objects etc.
   exec("./camera.cs");
   exec("./editor.cs");
   exec("./visitor.cs");
   exec("./items.cs");
   exec("./guide.cs");
   exec("./triggers.cs");
   exec("./markers.cs");
   exec("./museumEmp.cs");

}

function onServerDestroyed()
{
   // This function is called as part of a server shutdown.
}


//-----------------------------------------------------------------------

function onMissionLoaded()
{
   // Called by loadMission() once the mission is finished loading.
}

function onMissionEnded()
{
   // Called by endMission(), right before the mission is destroyed
}
```

```
//-----------------------------------------------------------------------
// Dealing with client connections
// These methods are extensions to the GameConnection class.
Extending
// GameConnection make is easier to deal with some of this
//functionality,
// but these could also be implemented as stand-alone functions.
//-----------------------------------------------------------------------


function GameConnection::onClientEnterGame(%this)
{
   // Every client get's a camera object.
   %this.camera = new Camera() {
      dataBlock = Observer;
   };
   MissionCleanup.add( %this.camera );
   %this.camera.scopeToClient(%this);

   // Create a player object.
   %spawnPoint = pickSpawnPoint();
   %this.createPlayer(%spawnPoint);




}

function GameConnection::onClientLeaveGame(%this)
{
   if (isObject(%this.camera))
      %this.camera.delete();
   if (isObject(%this.player))
      %this.player.delete();

}



function GameConnection::createPlayer(%this, %spawnPoint)
{
   if (%this.player > 0)  {
      // The client should not have a player currently
      // assigned.  Assigning a new one could result in
      // a player ghost.
      error( "Attempting to create an angus ghost!" );
   }

   // Create the player object
   %player = new Player() {
      dataBlock = PlayerShape;
      client = %this;
   };
   MissionCleanup.add(%player);

   // Player setup...
   %player.setTransform(%spawnPoint);
   %player.setShapeName(%this.name);

   // Update the camera to start with the player
```

```
   %this.camera.setTransform(%player.getEyeTransform());

   // Give the client control of the player
   %this.player = %player;
   %this.setControlObject(%player);

}


function pickSpawnPoint()
{
   // Pick the first object in drop point group and use it's
   // location as a spawn point.
   %group = nameToID("MissionGroup/PlayerDropPoints");
   if (%group != -1 && %group.getCount() != 0)
      return %group.getObject(0).getTransform();

   // If no object was found, return a point near the center of the
world
   error("Missing spawn point object and/or mission group " @
%groupName);
   return "0 0 300 1 0 0 0";
}
```

8.1.2 Museum\server\items.cs

```
//-----------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C
// Museum\server\items.cs
//-----------------------------------------------------------------------


datablock StaticShapeData(InfoDeskItem)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function InfoDeskItem::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowDeskInfo');
  }
}



datablock StaticShapeData(Base11Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base11Item::onCollision(%this, %obj, %col)
```

```
{
  if(%col.getClassName() $= "Player")
  {

    %client = %col.client;

    commandToClient(%client, 'ShowObjectInfo',11);
  }
}



datablock StaticShapeData(Base12Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base12Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowObjectInfo',12);
  }
}


datablock StaticShapeData(Base13Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base13Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowObjectInfo',13);
  }
}


datablock StaticShapeData(Base14Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base14Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowObjectInfo',14);
  }
}
```

```
datablock StaticShapeData(Base15Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base15Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowObjectInfo',15);
  }
}



datablock StaticShapeData(Base16Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base16Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowObjectInfo',16);
  }
}



datablock StaticShapeData(Base21Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base21Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowObjectInfo',21);
  }
}



datablock StaticShapeData(Base22Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
```

```
};


function Base22Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowObjectInfo',22);
  }
}



datablock StaticShapeData(Base23Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base23Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowObjectInfo',23);
  }
}


datablock StaticShapeData(Base24Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base24Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowObjectInfo',24);
  }
}



datablock StaticShapeData(Base25Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base25Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
```

```
    commandToClient(%client, 'ShowObjectInfo',25);
  }
}

datablock StaticShapeData(Base26Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base26Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowObjectInfo',26);
  }
}


datablock StaticShapeData(Base27Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base27Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowObjectInfo',27);
  }
}



datablock StaticShapeData(Base28Item)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/infoItemPack/infoItem.dts";
};


function Base28Item::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowObjectInfo',28);
  }
}




datablock StaticShapeData(SphereItem)
{
   category = "Static Shapes";
```

```
   shapeFile = "~/data/shapes/simpleshape/simpleshape1.dts";
};


function SphereItem::onCollision(%this, %obj, %col)
{
  if(%col.getClassName() $= "Player")
  {
    %client = %col.client;
    commandToClient(%client, 'ShowHealthKitInfo');
    %obj.schedule($HealthKitTimeoutValue -1000, "startFade", 1000, 0,
true);
    %obj.schedule($HealthKitTimeoutValue , "delete");
  }
}
```

8.1.3 Museum\server\triggers.cs

```
//-----------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C
// Museum\server\triggers.cs
//-----------------------------------------------------------------------


datablock TriggerData( ExpoOneTrigger )
{
      // tickPeriodMS is a value is used to control how often the
console
      // onTriggerTick callback is called while there are any objects
      // in the trigger.

      tickPeriodMS = 100;
};

function ExpoOneTrigger::onEnterTrigger( %this, %trigger, %obj )
{

     // This method is called whenever an object enters the %trigger
     // area, the object is passed as %obj.

     echo("obj " @ %obj.getId()@ "this " @ %this.getId());
     if (%obj.client.player == %obj)
     {
        MessagePopup("Ground Floor","Exhibition of sculptures",1000);

       PlaceTriggerControler.setText("Exhibition of sculptures");
       PlaceTriggerControler.setVisible(1);



        ExpoTriggerControler.schedule(0,"setText","       Ground
Floor  #   Exhibition of sculptures");
        ExpoTriggerControler.schedule(500,"setVisible",0);
        ExpoTriggerControler.schedule(2500,"setVisible",0);


// The default onEnterTrigger method (in the C++ code) invokes the
//::onTrigger(%trigger,1) method on
```

```
// method on every object (whatever it's type) in the same group as
//the trigger.


        Parent::onEnterTrigger( %this, %trigger, %obj );
     }
}

function ExpoOneTrigger::onLeaveTrigger( %this, %trigger, %obj )
{
      // This method is called whenever an object leaves the %trigger
      // area, the object is passed as %obj.

     if (%obj.client.player == %obj)
     {
       PlaceTriggerControler.setVisible(0);


/ The default onLeaveTriggermethod (in the C++ code) invokes the
//::onTrigger(%trigger,0)
// method on every object (whatever it's type) in the same group as
//the trigger.

       Parent::onLeaveTrigger( %this, %trigger, %obj );
     }
}


function ExpoOneTrigger::onTickTrigger( %this, %trigger )
{

    if (%obj.client.player == %obj)
    {
      // This method is called every tickPerioMS, as long as any
      // objects intersect the trigger.

// The default onTriggerTick method (in the C++ code) invokes the
//::onTriggerTick(%trigger)
// method on every object (whatever it's type) in the same group as
//the trigger.

      Parent::onEnterTrigger( %this, %trigger, %obj );
      //    %this.getNumObjects();
      //    %this.getObject(n);
    }

}




datablock TriggerData( ExpoTwoTrigger )
{

    tickPeriodMS = 100;
};

function ExpoTwoTrigger::onEnterTrigger( %this, %trigger, %obj )
{
    if (%obj.client.player == %obj)
```

```
    {
       MessagePopup("First Floor","Exhibition of paintings",1000);
       PlaceTriggerControler.setText("Exhibition of paintings");
       PlaceTriggerControler.setVisible(1);

       ExpoTriggerControler.schedule(0,"setText","        First floor
#   Exhibition of paintings");
       ExpoTriggerControler.schedule(500,"setVisible",0);
       ExpoTriggerControler.schedule(2500,"setVisible",0);

       Parent::onEnterTrigger( %this, %trigger, %obj );
    }
}


function ExpoTwoTrigger::onLeaveTrigger( %this, %trigger, %obj )
{
    if (%obj.client.player == %obj)
    {
      PlaceTriggerControler.setVisible(0);

      Parent::onLeaveTrigger( %this, %trigger, %obj );
    }
}


function ExpoTwoTrigger::onTickTrigger( %this, %trigger )
{
    if (%obj.client.player == %obj)
    {
       Parent::onEnterTrigger( %this, %trigger, %obj );
    }

}






datablock TriggerData( StartTrigger )
{
    tickPeriodMS = 100;
};


function StartTrigger::onEnterTrigger( %this, %trigger, %obj )
{
    PlaceTriggerControler.setVisible(0);
    helpMessage.setVisible(1);
    StartTriggerControler1.schedule(0,"setVisible",1);
    StartTriggerControler2.schedule(0,"setVisible",1);
    StartTriggerControler1.schedule(1500,"setVisible",0);
    StartTriggerControler2.schedule(1500,"setVisible",0);

    Parent::onEnterTrigger( %this, %trigger, %obj );
}

function StartTrigger::onLeaveTrigger( %this, %trigger, %obj )
{
     helpMessage.schedule(7000,"setVisible",0);
```

```
     Parent::onLeaveTrigger( %this, %trigger, %obj );

}

function StartTrigger::onTickTrigger( %this, %trigger )
{

     Parent::onEnterTrigger( %this, %trigger, %obj );

}



datablock TriggerData( InfoHouseTrigger )
{
     tickPeriodMS = 100;
};

function InfoHouseTrigger::onEnterTrigger( %this, %trigger, %obj )
{
     MessagePopup("Information office","Welcome to the information
office!",1500);
     InfoHouseTriggerControler.schedule(0,"setVisible",0);
     InfoHouseTriggerControler.schedule(2000,"setVisible",0);
     PlaceTriggerControler.setText("Information office");
     PlaceTriggerControler.setVisible(1);

     Parent::onEnterTrigger( %this, %trigger, %obj );
}

function InfoHouseTrigger::onLeaveTrigger( %this, %trigger, %obj )
{

      PlaceTriggerControler.setVisible(0);

      Parent::onLeaveTrigger( %this, %trigger, %obj );
}



datablock TriggerData( Object11Trigger )
{
      tickPeriodMS = 100;
};


function Object11Trigger::onEnterTrigger( %this, %trigger, %obj )
{
     echo(" obj " @ %obj.getId() @ " this " @ %this.getId()  @
"trigger" @ %trigger.getId() @ "||||||||||||||");

     if (%obj.client.player != %obj)
     {
         moreInfo();
         ObjectTriggerControler.schedule(0,"setText","
Sculpture11      #        Artist11");
         ObjectTriggerControler.schedule(1000,"setVisible",1);
         ObjectTriggerControler.schedule(5000,"setVisible",0);

         %obj.schedule(5000,"moveToNode",0);
```

```
         %obj.schedule(5800,"done");

         Parent::onEnterTrigger( %this, %trigger, %obj );
      }
}


datablock TriggerData( Object12Trigger )
{
      tickPeriodMS = 100;
};


function Object12Trigger::onEnterTrigger( %this, %trigger, %obj )
{
      echo(" obj " @ %obj.getId() @ " this " @ %this.getId() @
"trigger" @ %trigger.getId()  @ "|||||||");

      if (%obj.client.player != %obj)
      {
          moreInfo();
          ObjectTriggerControler.schedule(0,"setText","
Sculpture12      #        Artist12");

          ObjectTriggerControler.schedule(1000,"setVisible",1);
          ObjectTriggerControler.schedule(5000,"setVisible",0);

          %obj.schedule(5000,"moveToNode",0);
          %obj.schedule(5800,"done");

          Parent::onEnterTrigger( %this, %trigger, %obj );
      }
}


datablock TriggerData( Object13Trigger )
{
      tickPeriodMS = 100;
};


function Object13Trigger::onEnterTrigger( %this, %trigger, %obj )
{
       echo(" obj " @ %obj.getId() @ " this " @ %this.getId()  @
"trigger" @ %trigger.getId() @ "|||||||||||||||");

       if (%obj.client.player != %obj)
       {
          moreInfo();
          ObjectTriggerControler.schedule(0,"setText","
Sculpture13      #        Artist13");

          ObjectTriggerControler.schedule(1000,"setVisible",1);
          ObjectTriggerControler.schedule(5000,"setVisible",0);

          %obj.schedule(5000,"moveToNode",0);
          %obj.schedule(5800,"done");

          Parent::onEnterTrigger( %this, %trigger, %obj );
        }
}
```

```
datablock TriggerData( Object14Trigger )
{

      tickPeriodMS = 100;
};


function Object14Trigger::onEnterTrigger( %this, %trigger, %obj )
{
        echo(" obj " @ %obj.getId() @ " this " @ %this.getId() @
"trigger" @ %trigger.getId()  @ "|||||||");

        if (%obj.client.player != %obj)
        {
            moreInfo();
            ObjectTriggerControler.schedule(0,"setText","
Sculpture14      #        Artist14");

            ObjectTriggerControler.schedule(1000,"setVisible",1);
            ObjectTriggerControler.schedule(5000,"setVisible",0);

            %obj.schedule(5000,"moveToNode",0);
            %obj.schedule(5800,"done");

            Parent::onEnterTrigger( %this, %trigger, %obj );
         }
}


datablock TriggerData( Object15Trigger )
{

         tickPeriodMS = 100;
};


function Object15Trigger::onEnterTrigger( %this, %trigger, %obj )
{
         echo(" obj " @ %obj.getId() @ " this " @ %this.getId()  @
"trigger" @ %trigger.getId() @ "|||||||||||||||");

         if (%obj.client.player != %obj)
         {
             moreInfo();
             ObjectTriggerControler.schedule(0,"setText","
Sculpture15      #        Artist15");

             ObjectTriggerControler.schedule(1000,"setVisible",1);
             ObjectTriggerControler.schedule(5000,"setVisible",0);

             %obj.schedule(5000,"moveToNode",0);
             %obj.schedule(5800,"done");

             Parent::onEnterTrigger( %this, %trigger, %obj );
         }
}


datablock TriggerData( Object16Trigger )
```

```
{

      tickPeriodMS = 100;
};


function Object16Trigger::onEnterTrigger( %this, %trigger, %obj )
{
  echo(" obj " @ %obj.getId() @ " this " @ %this.getId() @ "trigger"
@ %trigger.getId()  @ "|||||||");

  if (%obj.client.player != %obj)
  {
    moreInfo();
    ObjectTriggerControler.schedule(0,"setText","      Sculpture16
#        Artist16");

    ObjectTriggerControler.schedule(1000,"setVisible",1);
    ObjectTriggerControler.schedule(5000,"setVisible",0);

    %obj.schedule(5000,"moveToNode",0);
    %obj.schedule(5800,"done");

    Parent::onEnterTrigger( %this, %trigger, %obj );
  }
}



datablock TriggerData( Object21Trigger )
{

      tickPeriodMS = 100;
};


function Object21Trigger::onEnterTrigger( %this, %trigger, %obj )
{
  echo(" obj " @ %obj.getId() @ " this " @ %this.getId() @ "trigger"
@ %trigger.getId()  @ "|||||||");

  if (%obj.client.player != %obj)
  {
    moreInfo();
    ObjectTriggerControler.schedule(0,"setText","      Painding21
#        Artist21");

    ObjectTriggerControler.schedule(1000,"setVisible",1);
    ObjectTriggerControler.schedule(5000,"setVisible",0);

    %obj.schedule(5000,"moveToNode",0);
    %obj.schedule(5800,"done");

    Parent::onEnterTrigger( %this, %trigger, %obj );
  }
}


datablock TriggerData( Object22Trigger )
{
```

```
      tickPeriodMS = 100;
};


function Object22Trigger::onEnterTrigger( %this, %trigger, %obj )
{
  echo(" obj " @ %obj.getId() @ " this " @ %this.getId() @ "trigger"
@ %trigger.getId()  @ "|||||||");

  if (%obj.client.player != %obj)
  {
    moreInfo();
    ObjectTriggerControler.schedule(0,"setText","      Painding22
#        Artist22");
    ObjectTriggerControler.schedule(1000,"setVisible",1);
    ObjectTriggerControler.schedule(5000,"setVisible",0);

    %obj.schedule(5000,"moveToNode",0);
    %obj.schedule(5800,"done");

    Parent::onEnterTrigger( %this, %trigger, %obj );
  }
}


datablock TriggerData( Object23Trigger )
{

      tickPeriodMS = 100;
};


function Object23Trigger::onEnterTrigger( %this, %trigger, %obj )
{
  echo(" obj " @ %obj.getId() @ " this " @ %this.getId() @ "trigger"
@ %trigger.getId()  @ "|||||||");

  if (%obj.client.player != %obj)
  {
    moreInfo();
    ObjectTriggerControler.schedule(0,"setText","      Painding23
#        Artist23");
    ObjectTriggerControler.schedule(1000,"setVisible",1);
    ObjectTriggerControler.schedule(5000,"setVisible",0);

    %obj.schedule(5000,"moveToNode",0);
    %obj.schedule(5800,"done");

    Parent::onEnterTrigger( %this, %trigger, %obj );
  }
}


datablock TriggerData( Object24Trigger )
{

      tickPeriodMS = 100;
};


function Object24Trigger::onEnterTrigger( %this, %trigger, %obj )
```

```
{
  echo(" obj " @ %obj.getId() @ " this " @ %this.getId() @ "trigger"
@ %trigger.getId()  @ "|||||||");

  if (%obj.client.player != %obj)
  {
    moreInfo();
    ObjectTriggerControler.schedule(0,"setText","      Painding24
#        Artist24");
    ObjectTriggerControler.schedule(1000,"setVisible",1);
    ObjectTriggerControler.schedule(5000,"setVisible",0);

    %obj.schedule(5000,"moveToNode",0);
    %obj.schedule(5800,"done");

    Parent::onEnterTrigger( %this, %trigger, %obj );
  }
}


datablock TriggerData( Object25Trigger )
{

      tickPeriodMS = 100;
};


function Object25Trigger::onEnterTrigger( %this, %trigger, %obj )
{
  echo(" obj " @ %obj.getId() @ " this " @ %this.getId() @ "trigger"
@ %trigger.getId()  @ "|||||||");

  if (%obj.client.player != %obj)
  {
    moreInfo();
    ObjectTriggerControler.schedule(0,"setText","      Painding25
#        Artist25");
    ObjectTriggerControler.schedule(1000,"setVisible",1);
    ObjectTriggerControler.schedule(5000,"setVisible",0);

    %obj.schedule(5000,"moveToNode",0);
    %obj.schedule(5800,"done");

    Parent::onEnterTrigger( %this, %trigger, %obj );
  }
}


datablock TriggerData( Object26Trigger )
{

      tickPeriodMS = 100;
};


function Object26Trigger::onEnterTrigger( %this, %trigger, %obj )
{
  echo(" obj " @ %obj.getId() @ " this " @ %this.getId() @ "trigger"
@ %trigger.getId()  @ "|||||||");

  if (%obj.client.player != %obj)
```

```
  {
    moreInfo();
    ObjectTriggerControler.schedule(0,"setText","      Painding26
#        Artist26");
    ObjectTriggerControler.schedule(1000,"setVisible",1);
    ObjectTriggerControler.schedule(5000,"setVisible",0);

    %obj.schedule(5000,"moveToNode",0);
    %obj.schedule(5800,"done");

    Parent::onEnterTrigger( %this, %trigger, %obj );
  }
}


datablock TriggerData( Object27Trigger )
{

      tickPeriodMS = 100;
};


function Object27Trigger::onEnterTrigger( %this, %trigger, %obj )
{
  echo(" obj " @ %obj.getId() @ " this " @ %this.getId() @ "trigger"
@ %trigger.getId()  @ "|||||||");

  if (%obj.client.player != %obj)
  {
    moreInfo();
    ObjectTriggerControler.schedule(0,"setText","      Painding27
#        Artist27");
    ObjectTriggerControler.schedule(1000,"setVisible",1);
    ObjectTriggerControler.schedule(5000,"setVisible",0);

    %obj.schedule(5000,"moveToNode",0);
    %obj.schedule(5800,"done");

    Parent::onEnterTrigger( %this, %trigger, %obj );
  }
}



datablock TriggerData( Object28Trigger )
{

      tickPeriodMS = 100;
};


function Object28Trigger::onEnterTrigger( %this, %trigger, %obj )
{
  echo(" obj " @ %obj.getId() @ " this " @ %this.getId() @ "trigger"
@ %trigger.getId()  @ "|||||||");

  if (%obj.client.player != %obj)
  {
    moreInfo();
    ObjectTriggerControler.schedule(0,"setText","      Painding28
#        Artist28");
```

```
    ObjectTriggerControler.schedule(1000,"setVisible",1);
    ObjectTriggerControler.schedule(5000,"setVisible",0);

    %obj.schedule(5000,"moveToNode",0);
    %obj.schedule(5800,"done");

    Parent::onEnterTrigger( %this, %trigger, %obj );
  }
}


function moreInfo()
{
    ArtistInfoTriggerControler.schedule(1000,"setVisible",1);
    ExhibitInfoTriggerControler.schedule(1000,"setVisible",1);
    ArtistInfoTriggerControler.schedule(5000,"setVisible",0);
    ExhibitInfoTriggerControler.schedule(5000,"setVisible",0);
}
```

8.1.4 Museum\server\visitor.cs

```
//-----------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C
// Museum\server\visitor.cs
//-----------------------------------------------------------------------



// Load dts shapes and merge animations
datablock TSShapeConstructor(PlayerDts)
{
   baseShape = "~/data/shapes/visitor/visitor.dts";
   sequence0 = "~/data/shapes/visitor/visitor_root.dsq root";
   sequence1 = "~/data/shapes/visitor/visitor_forward.dsq run";
   sequence2 = "~/data/shapes/visitor/visitor_back.dsq back";
   sequence3 = "~/data/shapes/visitor/visitor_side.dsq side";
   //sequence4 = "~/data/shapes/visitor/visitor_fall.dsq fall";
   sequence5 = "~/data/shapes/visitor/visitor_land.dsq land";
   sequence6 = "~/data/shapes/visitor/visitor_jump.dsq jump";
};

datablock PlayerData(PlayerShape)
{
   renderFirstPerson = false;
   shapeFile = "~/data/shapes/visitor/visitor.dts";
};


//-----------------------------------------------------------------------
// PlayerShape Datablock methods
//-----------------------------------------------------------------------

function PlayerShape::onAdd(%this,%obj)
{
   // Called when the PlayerData datablock is first 'read' by the
//engine (executable)
```

```
}

function PlayerShape::onRemove(%this, %obj)
{
   if (%obj.client.player == %obj)
      %obj.client.player = 0;
}

function PlayerShape::onNewDataBlock(%this,%obj)
{
   // Called when this PlayerData datablock is assigned to an object
}
```

8.1.5 Museum\server\guide.cs

```
//--------------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\server\guide.cs
//--------------------------------------------------------------------------


//--------------------------------------------------------------------------
// Demo Pathed AIPlayer.
//--------------------------------------------------------------------------

datablock TSShapeConstructor(GuideDts)
{
   baseShape = "~/data/shapes/guide/guide.dts";
   sequence0 = "~/data/shapes/guide/guide_root.dsq root";
   sequence1 = "~/data/shapes/guide/guide_forward.dsq run";
   sequence2 = "~/data/shapes/guide/guide_back.dsq back";
   sequence3 = "~/data/shapes/guide/guide_side.dsq side";

};

datablock PlayerData(GuidePlayer : PlayerShape)
{

   //renderFirstPerson = false;
   shapeFile = "~/data/shapes/guide/guide.dts";

};

function GuidePlayer::onReachDestination(%this,%obj)
{
   // Moves to the next node on the path.
     echo("onReachDestination");

}

function GuidePlayer::onEndOfPath(%this,%obj,%path)
{
   echo("onEndOfPath");

}
```

```
function GuidePlayer::onEndSequence(%this,%obj,%slot)
{
  echo("Sequence Done!");

}


//--------------------------------------------------------------------
// AIPlayer static functions
//--------------------------------------------------------------------

function AIPlayer::spawn(%name,%spawnPoint)
{
   // Create the demo player object
    echo("@@@@@@@@@@@@@@@spawn");
   %player = new AiPlayer() {
      dataBlock = GuidePlayer;
      path = "";
   };
   MissionCleanup.add(%player);
   %player.setShapeName(%name);
   %player.setTransform(%spawnPoint);

   return %player;
}


function AIPlayer::spawnOnPath(%name,%path)
{

   // Spawn a player and place him on the first node of the path
   echo("@@@@@@@@@@@@@@@spawnOnPath");
      %node = %path.getObject(0);
   %player = AIPlayer::spawn(%name,%node.getTransform());
    return %player;
}

//--------------------------------------------------------------------
// AIPlayer methods
//--------------------------------------------------------------------

function AIPlayer::followPath(%this,%path,%node)
{
   echo("@@@@@@@@@@@@@@@followPath");
   %this.path = %path;
   %this.targetNode = 1;
   %this.currentNode = 0 ;
   %this.moveToNode(1);
}




function AIPlayer::moveToNode(%this,%index)
{
    echo("@@@@@@@@@@@@@@@moveToNode" @ " " @ %this.path @ " " @
%index);
    // Move to the given path node index
    %this.currentNode = %index;
    %node = %this.path.getObject(%index);
```

```
    %this.setMoveDestination(%node.getTransform(), %index ==
%this.targetNode);
}


//-----------------------------------------------------------------

function AIPlayer::pushTask(%this,%method)
{
   if (%this.taskIndex $= "") {
      %this.taskIndex = 0;
      %this.taskCurrent = -1;
   }
   %this.task[%this.taskIndex] = %method;
   %this.taskIndex++;
   if (%this.taskCurrent == -1)
      %this.executeTask(%this.taskIndex - 1);
}

function AIPlayer::clearTasks(%this)
{
   %this.taskIndex = 0;
   %this.taskCurrent = -1;
}

function AIPlayer::nextTask(%this)
{
   if (%this.taskCurrent != -1)
      if (%this.taskCurrent < %this.taskIndex - 1)
         %this.executeTask(%this.taskCurrent++);
      else
         %this.taskCurrent = -1;
}

function AIPlayer::executeTask(%this,%index)
{
   %this.taskCurrent = %index;
   eval(%this.getId() @ "." @ %this.task[%index] @ ";");
}




function AIPlayer::wait(%this,%time)
{
   %this.schedule(%time * 1000,"nextTask");
}

function AIPlayer::done(%this,%time)
{
   %this.schedule(0,"delete");
}


function AIPlayer::animate(%this,%seq)
{
   %this.stopThread(0);
}


//-----------------------------------------------------------------
```

```
function AIManager::think(%this, %path)
{

   if (!isObject(%this.player))
      %this.player = %this.spawn(%path);
   %this.schedule(500,think);

}



function AIManager::spawn(%this, %choosepath)
{

    %player = AIPlayer::spawnOnPath("Kork",%choosepath);
    %player.followPath(%choosepath,-1);
    %player.schedule(10000,"delete");
    return %player;
}



function serverCmdGuideInformation(%client,%x)
{

    if(%x $= 11)
    {
       new ScriptObject(AIManager) {};
       MissionCleanup.add(AIManager);
       AIManager.think("MissionGroup/Paths/Path11");
       AIManager.schedule(5800,"delete");
    }

    else if(%x $= 12)
    {
       new ScriptObject(AIManager) {};
       MissionCleanup.add(AIManager);
       AIManager.think("MissionGroup/Paths/Path12");
       AIManager.schedule(5800,"delete");
    }

    else if(%x $= 13)
    {
       new ScriptObject(AIManager) {};
       MissionCleanup.add(AIManager);
       AIManager.think("MissionGroup/Paths/Path13");
       AIManager.schedule(5800,"delete");
    }

    else if(%x $= 14)
    {
       new ScriptObject(AIManager) {};
       MissionCleanup.add(AIManager);
       AIManager.think("MissionGroup/Paths/Path14");
       AIManager.schedule(5800,"delete");
    }

    else if(%x $= 15)
    {
```

```
   new ScriptObject(AIManager) {};
   MissionCleanup.add(AIManager);
   AIManager.think("MissionGroup/Paths/Path15");
   AIManager.schedule(5800,"delete");
 }

else if(%x $= 16)
{
   new ScriptObject(AIManager) {};
   MissionCleanup.add(AIManager);
   AIManager.think("MissionGroup/Paths/Path16");
   AIManager.schedule(5800,"delete");
}

else if(%x $= 21)
{
   new ScriptObject(AIManager) {};
   MissionCleanup.add(AIManager);
   AIManager.think("MissionGroup/Paths/Path21");
   AIManager.schedule(5800,"delete");
}

else if(%x $= 22)
{
   new ScriptObject(AIManager) {};
   MissionCleanup.add(AIManager);
   AIManager.think("MissionGroup/Paths/Path22");
   AIManager.schedule(5800,"delete");
}

else if(%x $= 23)
{
   new ScriptObject(AIManager) {};
   MissionCleanup.add(AIManager);
   AIManager.think("MissionGroup/Paths/Path23");
   AIManager.schedule(5800,"delete");
}

else if(%x $= 24)
{
   new ScriptObject(AIManager) {};
   MissionCleanup.add(AIManager);
   AIManager.think("MissionGroup/Paths/Path24");
   AIManager.schedule(5800,"delete");
}

else if(%x $= 25)
{
   new ScriptObject(AIManager) {};
   MissionCleanup.add(AIManager);
   AIManager.think("MissionGroup/Paths/Path25");
   AIManager.schedule(5800,"delete");
}

else if(%x $= 26)
{
   new ScriptObject(AIManager) {};
   MissionCleanup.add(AIManager);
   AIManager.think("MissionGroup/Paths/Path26");
   AIManager.schedule(5800,"delete");
}
```

```
    else if(%x $= 27)
    {
       new ScriptObject(AIManager) {};
       MissionCleanup.add(AIManager);
       AIManager.think("MissionGroup/Paths/Path27");
       AIManager.schedule(5800,"delete");
    }

    else if(%x $= 28)
    {
       new ScriptObject(AIManager) {};
       MissionCleanup.add(AIManager);
       AIManager.think("MissionGroup/Paths/Path28");
       AIManager.schedule(5800,"delete");
    }

}
```

8.1.6 Museum\server\museumEmp.cs

```
//--------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C
// Museum\server\museumEmp.cs
//--------------------------------------------------------------------


//$GuideTimeoutValue = 1000;

datablock StaticShapeData(Guide)
{
   category = "Static Shapes";
   shapeFile = "~/data/shapes/museumEmp/player.dts";
};
```

8.1.7 Museum\server\camera.cs

```
//--------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\server\camera.cs
//--------------------------------------------------------------------

// Global movement speed that affects all cameras.
$Camera::movementSpeed = 40;

//--------------------------------------------------------------------
// Define a datablock class to use for our observer camera
//--------------------------------------------------------------------

datablock CameraData(Observer)
```

```
{
   mode = "Observer";
};
```

8.1.8 Museum\server\defaults.cs

```
//-----------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\server\defaults.cs
//-----------------------------------------------------------------------


// List of master servers to query, each one is tried in order
// until one responds
$Pref::Server::RegionMask = 2;
$pref::Master[0] = "2:master.garagegames.com:28002";

// Information about the server
$Pref::Server::Name = "Torque TTB Server";
$Pref::Server::Info = "This is a Torque Game Engine Test Server.";

// The connection error message is transmitted to the client
immediatly
// on connection, if any further error occures during the connection
// process, such as network traffic mismatch, or missing files, this
error
// message is display. This message should be replaced with
information
// usefull to the client, such as the url or ftp address of where the
// latest version of the game can be obtained.
$Pref::Server::ConnectionError =
   "You do not have the correct version of the Torque Game Engine or
"@
   "the related art needed to connect to this server, please contact
"@
   "the server operator to obtain the latest version of this game.";

// The network port is also defined by the client, this value
// overrides pref::net::port for dedicated servers
$Pref::Server::Port = 28000;

// If the password is set, clients must provide it in order
// to connect to the server
$Pref::Server::Password = "";

// Password for admin clients
$Pref::Server::AdminPassword = "";

// Misc server settings.
$Pref::Server::MaxPlayers = 64;
$Pref::Server::FloodProtectionEnabled = 1;
$Pref::Server::MaxChatLen = 120;
```

8.1.9 Museum\server\editor.cs

```
//-----------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\server\editor.cs
//-----------------------------------------------------------------


// This file contains shape declarations and functions used by the
//mission
// editor. The mission editor invokes datablock create methods when
//it
// wants to create an object of that type.

// Declare a static shape create method.  This allows DTS objects to
//be loaded
// using the StaticShape simulation object.

function StaticShapeData::create(%data)
{
   %obj = new StaticShape() {
      dataBlock = %data;
   };
   return %obj;
}

// Declare two marker objects used to place waypoints

datablock MissionMarkerData(WayPointMarker)
{
   category = "Misc";
   shapeFile = "~/data/shapes/markers/octahedron.dts";
};

datablock MissionMarkerData(SpawnSphereMarker)
{
   category = "Misc";
   shapeFile = "~/data/shapes/markers/octahedron.dts";
};

function MissionMarkerData::create(%block)
{
   switch$(%block)
   {
      case "WayPointMarker":
         %obj = new WayPoint() {
            dataBlock = %block;
         };
         return(%obj);
      case "SpawnSphereMarker":
         %obj = new SpawnSphere() {
            datablock = %block;
         };
         return(%obj);
   }
   return(-1);
}


//-----------------------------------------------------------------
```

```
// Misc. server commands used for editing
//-----------------------------------------------------------------------

function serverCmdToggleCamera(%client)
{
   if ($Server::ServerType $= "SinglePlayer") {
      %control = %client.getControlObject();
      if (%control == %client.player) {
         %control = %client.camera;
         %control.mode = toggleCameraFly;
      }
      else {
         %control = %client.player;
         %control.mode = observerFly;
      }
      %client.setControlObject(%control);
   }
}

function serverCmdDropPlayerAtCamera(%client)
{
   if (!%client.player.isMounted())
      %client.player.setTransform(%client.camera.getTransform());
   %client.player.setVelocity("0 0 0");
   %client.setControlObject(%client.player);
}

function serverCmdDropCameraAtPlayer(%client)
{
   %client.camera.setTransform(%client.player.getEyeTransform());
   %client.camera.setVelocity("0 0 0");
   %client.setControlObject(%client.camera);
}


function dropFreakinCameraAtPlayer()
{
   $dropcameracount++;
   %cl = ClientGroup.getObject(0);
   if (%cl.camera) dropCameraAtPlayer(1);
   else if ($dropcameracount<100)
schedule(100,0,dropFreakinCameraAtPlayer);
}
```

8.1.10 Museum\server\markers.cs

```
//-----------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C
// Museum\server\markers.cs
//-----------------------------------------------------------------------


datablock MissionMarkerData(WayPointMarker)
{
   category = "Misc";
   shapeFile = "~/data/shapes/markers/octahedron.dts";
};
```

```
datablock MissionMarkerData(SpawnSphereMarker)
{
   category = "Misc";
   shapeFile = "~/data/shapes/markers/octahedron.dts";
};


// - serveral marker types may share MissionMarker datablock type
function MissionMarkerData::create(%block)
{
   switch$(%block)
   {
      case "WayPointMarker":
         %obj = new WayPoint() {
            dataBlock = %block;
         };
         return(%obj);
      case "SpawnSphereMarker":
         %obj = new SpawnSphere() {
            datablock = %block;
         };
         return(%obj);
   }
   return(-1);
}
//--------------------------------------------------------------------------
```

8.2 Πελάτης (client).

8.2.1 Museum\client\clientGui.cs

```
//--------------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\client\clientGui.cs
//--------------------------------------------------------------------------


//--------------------------------------------------------------------------
// PlayGui is the main TSControl through which the game is viewed.
//--------------------------------------------------------------------------

function PlayGui::onWake(%this)
{
   // Turn off any shell sounds...
   // alxStop( ... );
   $enableDirectInput = "1";
   activateDirectInput();

   // Activate the game's action map
   moveMap.push();
}

function PlayGui::onSleep(%this)
```

```
{
   // Pop the keymap
   moveMap.pop();
}
```

8.2.2 Museum\client\loadingGui.cs

```
//--------------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\client\loadingGui.cs
//--------------------------------------------------------------------------


function LoadingGui::onAdd(%this)
{
   %this.qLineCount = 0;
}

function LoadingGui::onWake(%this)
{
   // Play sound...
   CloseMessagePopup();
}

function LoadingGui::onSleep(%this)
{
   // Clear the load info:
   if ( %this.qLineCount !$= "" )
   {
      for ( %line = 0; %line < %this.qLineCount; %line++ )
         %this.qLine[%line] = "";
   }
   %this.qLineCount = 0;

   LOAD_MapName.setText( "" );
   LOAD_MapDescription.setText( "" );
   LoadingProgress.setValue( 0 );
   LoadingProgressTxt.setValue( "WAITING FOR SERVER" );


}
```

8.2.3 Museum\client\serverConnection.cs

```
//--------------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\client\serverConnection.cs
//--------------------------------------------------------------------------



// Functions dealing with connecting to a server
```

```
//---------------------------------------------------------------------
// Server messages
//---------------------------------------------------------------------

function onServerMessage(%mesg)
{
   // Server message text which includes chat from other players.
}


//---------------------------------------------------------------------
// Server connection error
//---------------------------------------------------------------------

addMessageCallback( 'MsgConnectionError',
handleConnectionErrorMessage );

function handleConnectionErrorMessage(%msgType, %msgString,
%msgError)
{
   // On connect the server transmits a message to display if there
   // are any problems with the connection.  Most connection errors
   // are game version differences, so hopefully the server message
   // will tell us where to get the latest version of the game.
   $ServerConnectionErrorMessage = %msgError;
}


//---------------------------------------------------------------------
// GameConnection client callbacks
//---------------------------------------------------------------------

function GameConnection::initialControlSet(%this)
{
   echo ("*** Initial Control Object");

   // The first control object has been set by the server
   // and we are now ready to go.

   // first check if the editor is active
   if (!Editor::checkActiveLoadDone())
   {
      if (Canvas.getContent() != PlayGui.getId())
         Canvas.setContent(PlayGui);
   }
}

function GameConnection::setLagIcon(%this, %state)
{
   if (%this.getAddress() $= "local")
      return;
   // Called when the server lag state changes state = true/false
}

function GameConnection::onConnectionAccepted(%this)
{
   // Called on the new connection object after connect() succeeds.
}

function GameConnection::onConnectionTimedOut(%this)
{
```

```
   // Called when an established connection times out
   disconnectedCleanup();
   MessageBoxOK( "TIMED OUT", "The server connection has timed
out.");
}

function GameConnection::onConnectionDropped(%this, %msg)
{
   // Established connection was dropped by the server
   disconnectedCleanup();
   MessageBoxOK( "DISCONNECT", "The server has dropped the
connection: " @ %msg);
}

function GameConnection::onConnectionError(%this, %msg)
{
   // General connection error, usually raised by ghosted objects
   // initialization problems, such as missing files.  We'll display
   // the server's connection error message.
   disconnectedCleanup();
   MessageBoxOK( "DISCONNECT", $ServerConnectionErrorMessage @ " (" @
%msg @ ")" );
}


//--------------------------------------------------------------------------
// Connection Failed Events
//--------------------------------------------------------------------------

function GameConnection::onConnectRequestRejected( %this, %msg )
{
   switch$(%msg)
   {
      case "CR_INVALID_PROTOCOL_VERSION":
         %error = "Incompatible protocol version: Your game version
is not compatible with this server.";
      case "CR_INVALID_CONNECT_PACKET":
         %error = "Internal Error: badly formed network packet";
      case "CR_YOUAREBANNED":
         %error = "You are not allowed to play on this server.";
      case "CR_SERVERFULL":
         %error = "This server is full.";
      case "CHR_PASSWORD":
         // XXX Should put up a password-entry dialog.
         if ($Client::Password $= "")
            MessageBoxOK( "REJECTED", "That server requires a
password.");
         else {
            $Client::Password = "";
            MessageBoxOK( "REJECTED", "That password is incorrect.");
         }
         return;
      case "CHR_PROTOCOL":
         %error = "Incompatible protocol version: Your game version
is not compatible with this server.";
      case "CHR_CLASSCRC":
         %error = "Incompatible game classes: Your game version is
not compatible with this server.";
      case "CHR_INVALID_CHALLENGE_PACKET":
         %error = "Internal Error: Invalid server response packet";
      default:
```

```
         %error = "Connection error.  Please try another server.
Error code: (" @ %msg @ ")";
   }
   disconnectedCleanup();
   MessageBoxOK( "REJECTED", %error);
}

function GameConnection::onConnectRequestTimedOut(%this)
{
   disconnectedCleanup();
   MessageBoxOK( "TIMED OUT", "Your connection to the server timed
out." );
}


//---------------------------------------------------------------------
// Disconnect
//---------------------------------------------------------------------

function disconnect()
{
   // Delete the connection if it's still there.
   if (isObject(ServerConnection))
      ServerConnection.delete();
   disconnectedCleanup();

   // Call destroyServer in case we're hosting
   destroyServer();
}

function disconnectedCleanup()
{
   // Terminate all playing sounds
   alxStopAll();
   if (isObject(MusicPlayer))
      MusicPlayer.stop();

   // Back to the launch screen
   Canvas.setContent(MainMenuGui);

   // Dump anything we're not using
   clearTextureHolds();
   purgeResources();
}
```

8.2.4 Museum\client\defaultBind.cs

```
//---------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\client\defaultBind.cs
//---------------------------------------------------------------------



//---------------------------------------------------------------------
```

```
// Key bindings for the in-game action Map
//------------------------------------------------------------------------

if ( isObject( moveMap ) )
   moveMap.delete();
new ActionMap(moveMap);

// Misc. in-game keys

function escapeFromGame()
{
   if ( $Server::ServerType $= "SinglePlayer" )
      MessageBoxYesNo( "Quit Mission", "Exit from this Mission?",
"disconnect();", "");
   else
      MessageBoxYesNo( "Disconnect", "Disconnect from the server?",
"disconnect();", "");
}

moveMap.bindCmd(keyboard, "escape", "", "escapeFromGame();");


function displayHelpMessage()
{

MessageBoxOK("Help","To move the visitor use the following buttons:\n
w:forword \n s:backwards \n a:left \n d:right \n\n To return to main
menu push the escape button");

}

moveMap.bindCmd(keyboard, "h", "", "displayHelpMessage();");



// Movement Keys

$movementSpeed = 1; // m/s

function setSpeed(%speed)
{
   if(%speed)
      $movementSpeed = %speed;
}

function moveleft(%val)
{
   $mvLeftAction = %val * $movementSpeed;
}

function moveright(%val)
{
   $mvRightAction = %val * $movementSpeed;
}

function moveforward(%val)
{
   $mvForwardAction = %val * $movementSpeed;
}

function movebackward(%val)
```

```
{
   $mvBackwardAction = %val * $movementSpeed;
}

function moveup(%val)
{
   $mvUpAction = %val * $movementSpeed;
}

function movedown(%val)
{
   $mvDownAction = %val * $movementSpeed;
}

function turnLeft( %val )
{
   $mvYawRightSpeed = %val ? $Pref::Input::KeyboardTurnSpeed : 0;
}

function turnRight( %val )
{
   $mvYawLeftSpeed = %val ? $Pref::Input::KeyboardTurnSpeed : 0;
}

function panUp( %val )
{
   $mvPitchDownSpeed = %val ? $Pref::Input::KeyboardTurnSpeed : 0;
}

function panDown( %val )
{
   $mvPitchUpSpeed = %val ? $Pref::Input::KeyboardTurnSpeed : 0;
}

function getMouseAdjustAmount(%val)
{
   // based on a default camera fov of 90'
   return(%val * ($cameraFov / 90) * 0.01);
}

function yaw(%val)
{
   $mvYaw += getMouseAdjustAmount(%val);
}

function pitch(%val)
{
   $mvPitch += getMouseAdjustAmount(%val);
}

function jump(%val)
{
   $mvTriggerCount2++;
}

function mouseTrigger(%val)
{
   $mvTriggerCount0++;
}

moveMap.bind( keyboard, a, moveleft );
```

```
moveMap.bind( keyboard, d, moveright );
moveMap.bind( keyboard, w, moveforward );
moveMap.bind( keyboard, s, movebackward );
moveMap.bind( keyboard, space, jump );


moveMap.bind( mouse, xaxis, yaw );
moveMap.bind( mouse, yaxis, pitch );
moveMap.bind( mouse, button0, mouseTrigger );

// Camera & View functions

function toggleFreeLook( %val )
{
   if ( %val )
      $mvFreeLook = true;
   else
      $mvFreeLook = false;
}

$firstPerson = true;
function toggleFirstPerson(%val)
{
   if (%val)
   {
      $firstPerson = !$firstPerson;
      ServerConnection.setFirstPerson($firstPerson);
   }
}

function toggleCamera(%val)
{
   if (%val)
      commandToServer('ToggleCamera');
}

moveMap.bind( keyboard, z, toggleFreeLook );
moveMap.bind(keyboard, tab, toggleFirstPerson );
moveMap.bind(keyboard, "alt c", toggleCamera);


// Helper functions used by the mission editor

function dropCameraAtPlayer(%val)
{
   if (%val)
      commandToServer('dropCameraAtPlayer');
}

function dropPlayerAtCamera(%val)
{
   if (%val)
      commandToServer('DropPlayerAtCamera');
}

moveMap.bind(keyboard, "F8", dropCameraAtPlayer);
moveMap.bind(keyboard, "F7", dropPlayerAtCamera);


//------------------------------------------------------------------------
// Key bindings for the Global action map available everywhere
```

```
//---------------------------------------------------------------------

// Misc.

GlobalActionMap.bind(keyboard, "tilde", toggleConsole);
GlobalActionMap.bindCmd(keyboard, "alt enter", "",
"toggleFullScreen();");


// Dubuging Functions

$MFDebugRenderMode = 0;
function cycleDebugRenderMode(%val)
{
   if (!%val)
      return;

   if (getBuildString() $= "Debug")
   {
      if($MFDebugRenderMode == 0)
      {
         // Outline mode, including fonts so no stats
         $MFDebugRenderMode = 1;
         GLEnableOutline(true);
      }
      else if ($MFDebugRenderMode == 1)
      {
         // Interior debug mode
         $MFDebugRenderMode = 2;
         GLEnableOutline(false);
         setInteriorRenderMode(7);
         showInterior();
      }
      else if ($MFDebugRenderMode == 2)
      {
         // Back to normal
         $MFDebugRenderMode = 0;
         setInteriorRenderMode(0);
         GLEnableOutline(false);
         show();
      }
   }
   else
   {
      echo("Debug render modes only available when running a Debug
build.");
   }
}

GlobalActionMap.bind(keyboard, "F9", cycleDebugRenderMode);

//---------------------------------------------------------------------------------------
```

8.2.5 Museum\client\missionDownload.cs

```
//---------------------------------------------------------------------
// Mordern Art Museum
```

```
// Created by Antonopoulou C.
// Museum\client\missionDownload.cs
//---------------------------------------------------------------------

// Mission Loading & Mission Info
// The mission loading server handshaking is handled by the
// common/client/missingLoading.cs.  This portion handles the
interface
// with the game GUI.


// Loading Phases:
// Phase 1: Download Datablocks
// Phase 2: Download Ghost Objects
// Phase 3: Scene Lighting

//---------------------------------------------------------------------
// Phase 1
//---------------------------------------------------------------------

function onMissionDownloadPhase1(%missionName, %musicTrack)
{
   // Close and clear the message hud (in case it's open)
   //cls();

   // Reset the loading progress controls:
   LoadingProgress.setValue(0);
   LoadingProgressTxt.setValue("LOADING DATABLOCKS");
}

function onPhase1Progress(%progress)
{
   LoadingProgress.setValue(%progress);
   Canvas.repaint();
}

function onPhase1Complete()
{
}

//---------------------------------------------------------------------
// Phase 2
//---------------------------------------------------------------------

function onMissionDownloadPhase2()
{
   // Reset the loading progress controls:
   LoadingProgress.setValue(0);
   LoadingProgressTxt.setValue("LOADING OBJECTS");
   Canvas.repaint();
}

function onPhase2Progress(%progress)
{
   LoadingProgress.setValue(%progress);
   Canvas.repaint();
}

function onPhase2Complete()
{
}
```

```
//--------------------------------------------------------------------------
// Phase 3
//--------------------------------------------------------------------------

function onMissionDownloadPhase3()
{
   LoadingProgress.setValue(0);
   LoadingProgressTxt.setValue("LIGHTING MISSION");
   Canvas.repaint();
}

function onPhase3Progress(%progress)
{
   LoadingProgress.setValue(%progress);
}

function onPhase3Complete()
{
   LoadingProgress.setValue( 1 );
   $lightingMission = false;
}

//--------------------------------------------------------------------------
// Mission loading done
//--------------------------------------------------------------------------

function onMissionDownloadComplete()
{
   // Client will shortly be dropped into the game, so this is
   // good place for any last minute gui cleanup.
}


//--------------------------------------------------------------------------
// Before downloading a mission, the server transmits the mission
// information through these messages.
//--------------------------------------------------------------------------

addMessageCallback( 'MsgLoadInfo', handleLoadInfoMessage );
addMessageCallback( 'MsgLoadDescripition',
handleLoadDescriptionMessage );
addMessageCallback( 'MsgLoadInfoDone', handleLoadInfoDoneMessage );


function handleLoadInfoMessage( %msgType, %msgString, %mapName ) {

      // Need to pop up the loading gui to display this stuff.
      Canvas.setContent("LoadingGui");

      // Clear all of the loading info lines:
      for( %line = 0; %line < LoadingGui.qLineCount; %line++ )
            LoadingGui.qLine[%line] = "";
      LoadingGui.qLineCount = 0;

   //
      LOAD_MapName.setText( %mapName );
}


function handleLoadDescriptionMessage( %msgType, %msgString, %line )
```

```
{
      LoadingGui.qLine[LoadingGui.qLineCount] = %line;
      LoadingGui.qLineCount++;

   // Gather up all the previous lines, append the current one
   // and stuff it into the control
      %text = "<spush><font:Arial:16>";

      for( %line = 0; %line < LoadingGui.qLineCount - 1; %line++ )
            %text = %text @ LoadingGui.qLine[%line] @ " ";
   %text = %text @ LoadingGui.qLine[%line] @ "<spop>";

      LOAD_MapDescription.setText( %text );
}


function handleLoadInfoDoneMessage( %msgType, %msgString )
{
   // This will get called after the last description line is sent.
}
```

8.2.6 Museum\client\clientCommands

```
//--------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\client\clientCommands
//--------------------------------------------------------------------


function clientCmdShowDeskInfo()
{
  MessageBoxYesNo("Information Desk!","Welcome to the Information
desk! \n Would you like some info?","displayDeskInfo();");

}

function displayDeskInfo()
{
 MessageBoxOK("General Information","There are two exhibitions in the
museum:\n 1. Exhibition of sculptures \n 2. Exhibition of
paintings");
}



function clientCmdShowObjectInfo(%ob)
{
  if(%ob $=11)
      MessageBoxYesNo(" Exhibition of sculptures-Sculpture1", "
Display info for this object? ","displayInfo(11);");
  else if(%ob $=12)
      MessageBoxYesNo(" Exhibition of sculptures-Sculpture2", "
Display info for this object? ","displayInfo(12);");
  else if(%ob $=13)
```

```
      MessageBoxYesNo(" Exhibition of sculptures-Sculpture3", "
Display info for this object? ","displayInfo(13);");
  else if(%ob $=14)
      MessageBoxYesNo(" Exhibition of sculptures-Sculpture4", "
Display info for this object? ","displayInfo(14);");
  else if(%ob $=15)
      MessageBoxYesNo(" Exhibition of sculptures-Sculpture5", "
Display info for this object? ","displayInfo(15);");
  else if(%ob $=16)
      MessageBoxYesNo(" Exhibition of sculptures-Sculpture6", "
Display info for this object? ","displayInfo(16);");
  else if(%ob $=21)
      MessageBoxYesNo(" Exhibition of paintings-Painting1", " Display
info for this object? ","displayInfo(21);");
  else if(%ob $=22)
      MessageBoxYesNo(" Exhibition of paintings-Painting2", " Display
info for this object? ","displayInfo(22);");
  else if(%ob $=23)
      MessageBoxYesNo(" Exhibition of paintings-Painting3", " Display
info for this object? ","displayInfo(23);");
  else if(%ob $=24)
      MessageBoxYesNo(" Exhibition of paintings-Painting4", " Display
info for this object? ","displayInfo(24);");
  else if(%ob $=25)
      MessageBoxYesNo(" Exhibition of paintings-Painting5", " Display
info for this object? ","displayInfo(25);");
  else if(%ob $=26)
      MessageBoxYesNo(" Exhibition of paintings-Painting6", " Display
info for this object? ","displayInfo(26);");
  else if(%ob $=27)
      MessageBoxYesNo(" Exhibition of paintings-Painting7", " Display
info for this object? ","displayInfo(27);");
  else if(%ob $=28)
      MessageBoxYesNo(" Exhibition of paintings-Painting8", " Display
info for this object? ","displayInfo(28);");
}


function displayInfo(%ob)
{

   if(%ob $=11)
   {
     MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Sculpture1 \n Category: Sculpture \n Artist: Artist11 \n \nWould you
like more information from a guide?","GuideInfo(11);");
   }

   else if(%ob $=12)
   {
     MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Sculpture2 \n Category: Sculpture \n Artist: Artist12\n \nWould you
like more information from a guide?","GuideInfo(12);");
   }

   else if(%ob $=13)
   {
     MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Sculpture3 \n Category: Sculpture \n Artist: Artist13\n \nWould you
like more information from a guide?","GuideInfo(13);");
   }
```

```
   else if(%ob $=14)
   {
     MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Sculpture4 \n Category: Sculpture \n Artist: Artist14\n \nWould you
like more information from a guide?","GuideInfo(14);");
   }

   else if(%ob $=15)
   {
      MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Sculpture5 \n Category: Sculpture \n Artist: Artist15\n \nWould you
like more information from a guide?","GuideInfo(15);");
   }

   else if(%ob $=16)
   {
      MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Sculpture6 \n Category: Sculpture \n Artist: Artist16\n \nWould you
like more information from a guide?","GuideInfo(16);");
   }

   else if(%ob $=21)
   {
       MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Painting1 \n Category: Painting \n Artist: Artist21\n \nWould you
like more information from a guide?","GuideInfo(21);");
   }

   else if(%ob $=22)
   {
       MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Painting2 \n Category: Painting \n Artist: Artist22\n \nWould you
like more information from a guide?","GuideInfo(22);");
   }

   else if(%ob $=23)
   {
       MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Painting3 \n Category: Painting \n Artist: Artist23\n \nWould you
like more information from a guide?","GuideInfo(23);");
   }

   else if(%ob $=24)
   {
       MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Painting4 \n Category: Painting \n Artist: Artist24\n \nWould you
like more information from a guide?","GuideInfo(24);");
   }

   else if(%ob $=25)
   {
       MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Painting5 \n Category: Painting \n Artist: Artist25\n \nWould you
like more information from a guide?","GuideInfo(25);");
   }

   else if(%ob $=26)
   {
```

```
       MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Painting6 \n Category: Painting \n Artist: Artist26\n \nWould you
like more information from a guide?","GuideInfo(26);");
   }

   else if(%ob $=27)
   {
       MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Painting7 \n Category: Painting \n Artist: Artist27\n \nWould you
like more information from a guide?","GuideInfo(27);");
   }

   else if(%ob $=28)
   {
       MessageBoxYesNo("Exhibit's Information ","Name of Exhibit:
Painting8 \n Category: Painting \n Artist: Artist28\n \nWould you
like more information from a guide?","GuideInfo(28);");
   }

}


function GuideInfo(%y)
{
  echo("ClientGuideInfo");
  CommandToServer('GuideInformation',%y);
}
```

8.2.7 Museum\client\defaults.cs

```
//-----------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\client\defaults.cs
//-----------------------------------------------------------------------



// The master server is declared with the server defaults, which is
// loaded on both clients & dedicated servers.  If the server mod
// is not loaded on a client, then the master must be defined.
//$pref::Master[0] = "2:v12master.dyndns.org:28002";

$pref::Player::Name = "Test Guy";
$pref::Player::defaultFov = 90;
$pref::Player::zoomSpeed = 0;

$pref::Net::LagThreshold = 400;

$pref::shadows = "2";
$pref::HudMessageLogSize = 40;
$pref::ChatHudLength = 1;

$pref::Input::LinkMouseSensitivity = 1;
// DInput keyboard, mouse, and joystick prefs
$pref::Input::KeyboardEnabled = 1;
$pref::Input::MouseEnabled = 1;
$pref::Input::JoystickEnabled = 0;
```

```
$pref::Input::KeyboardTurnSpeed = 0.1;

$pref::sceneLighting::cacheSize = 20000;
$pref::sceneLighting::purgeMethod = "lastCreated";
$pref::sceneLighting::cacheLighting = 1;
$pref::sceneLighting::terrainGenerateLevel = 1;

$pref::Terrain::DynamicLights = 1;
$pref::Interior::TexturedFog = 0;

$pref::Video::displayDevice = "OpenGL";
$pref::Video::allowOpenGL = 1;
$pref::Video::allowD3D = 1;
$pref::Video::preferOpenGL = 1;
$pref::Video::appliedPref = 0;
$pref::Video::disableVerticalSync = 1;
$pref::Video::monitorNum = 0;
$pref::Video::windowedRes = "800 600";
$pref::Video::screenShotFormat = "PNG";

$pref::OpenGL::force16BitTexture = "0";
$pref::OpenGL::forcePalettedTexture = "0";
$pref::OpenGL::maxHardwareLights = 3;
$pref::VisibleDistanceMod = 1.0;
```

8.2.8 Museum\client\optionsDlg.cs

```
//--------------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\client\optionsDlg.cs
//--------------------------------------------------------------------------


function optionsDlg::setPane(%this, %pane)
{
   OptAudioPane.setVisible(false);
   OptGraphicsPane.setVisible(false);
   OptNetworkPane.setVisible(false);
   ("Opt" @ %pane @ "Pane").setVisible(true);
}

function OptionsDlg::onWake(%this)
{
   OptGraphicsButton.performClick();
   %buffer = getDisplayDeviceList();
   %count = getFieldCount( %buffer );
   OptGraphicsDriverMenu.clear();
   OptScreenshotMenu.init();
   OptScreenshotMenu.setValue($pref::Video::screenShotFormat);
   for(%i = 0; %i < %count; %i++)
      OptGraphicsDriverMenu.add(getField(%buffer, %i), %i);
   %selId = OptGraphicsDriverMenu.findText(
$pref::Video::displayDevice );
      if ( %selId == -1 )
            %selId = 0; // How did THAT happen?
      OptGraphicsDriverMenu.setSelected( %selId );
```

```
      OptGraphicsDriverMenu.onSelect( %selId, "" );

   // Audio
   OptAudioUpdate();
   OptAudioVolumeMaster.setValue($pref::Audio::masterVolume);
   OptAudioVolumeShell.setValue(
$pref::Audio::channelVolume[$GuiAudioType]);
   OptAudioVolumeSim.setValue(
$pref::Audio::channelVolume[$SimAudioType]);
   OptAudioDriverList.clear();
   OptAudioDriverList.add("OpenAL", 1);
   OptAudioDriverList.add("none", 2);
   %selId = OptAudioDriverList.findText($pref::Audio::driver);
      if ( %selId == -1 )
            %selId = 0; // How did THAT happen?
      OptAudioDriverList.setSelected( %selId );
      OptAudioDriverList.onSelect( %selId, "" );
}

function OptionsDlg::onSleep(%this)
{
}

function OptGraphicsDriverMenu::onSelect( %this, %id, %text )
{
      // Attempt to keep the same res and bpp settings:
      if ( OptGraphicsResolutionMenu.size() > 0 )
            %prevRes = OptGraphicsResolutionMenu.getText();
      else
            %prevRes = getWords( $pref::Video::resolution, 0, 1 );

      // Check if this device is full-screen only:
      if ( isDeviceFullScreenOnly( %this.getText() ) )
      {
            OptGraphicsFullscreenToggle.setValue( true );
            OptGraphicsFullscreenToggle.setActive( false );
            OptGraphicsFullscreenToggle.onAction();
      }
      else
            OptGraphicsFullscreenToggle.setActive( true );

      if ( OptGraphicsFullscreenToggle.getValue() )
      {
            if ( OptGraphicsBPPMenu.size() > 0 )
                  %prevBPP = OptGraphicsBPPMenu.getText();
            else
                  %prevBPP = getWord( $pref::Video::resolution, 2 );
      }

      // Fill the resolution and bit depth lists:
      OptGraphicsResolutionMenu.init( %this.getText(),
OptGraphicsFullscreenToggle.getValue() );
      OptGraphicsBPPMenu.init( %this.getText() );

      // Try to select the previous settings:
      %selId = OptGraphicsResolutionMenu.findText( %prevRes );
      if ( %selId == -1 )
            %selId = 0;
      OptGraphicsResolutionMenu.setSelected( %selId );

      if ( OptGraphicsFullscreenToggle.getValue() )
```

```
      {
            %selId = OptGraphicsBPPMenu.findText( %prevBPP );
            if ( %selId == -1 )
                  %selId = 0;
            OptGraphicsBPPMenu.setSelected( %selId );
            OptGraphicsBPPMenu.setText(
OptGraphicsBPPMenu.getTextById( %selId ) );
      }
      else
            OptGraphicsBPPMenu.setText( "Default" );

}

function OptGraphicsResolutionMenu::init( %this, %device, %fullScreen
)
{
      %this.clear();
      %resList = getResolutionList( %device );
      %resCount = getFieldCount( %resList );
      %deskRes = getDesktopResolution();

   %count = 0;
      for ( %i = 0; %i < %resCount; %i++ )
      {
            %res = getWords( getField( %resList, %i ), 0, 1 );

            if ( !%fullScreen )
            {
                  if ( firstWord( %res ) >= firstWord( %deskRes ) )
                        continue;
                  if ( getWord( %res, 1 ) >= getWord( %deskRes, 1 ) )
                        continue;
            }

            // Only add to list if it isn't there already:
            if ( %this.findText( %res ) == -1 )
      {
                  %this.add( %res, %count );
         %count++;
      }
      }
}

function OptGraphicsFullscreenToggle::onAction(%this)
{
   Parent::onAction();
   %prevRes = OptGraphicsResolutionMenu.getText();

   // Update the resolution menu with the new options
   OptGraphicsResolutionMenu.init( OptGraphicsDriverMenu.getText(),
%this.getValue() );

   // Set it back to the previous resolution if the new mode supports
it.
   %selId = OptGraphicsResolutionMenu.findText( %prevRes );
   if ( %selId == -1 )
      %selId = 0;
      OptGraphicsResolutionMenu.setSelected( %selId );
}
```

```
function OptGraphicsBPPMenu::init( %this, %device )
{
      %this.clear();

      if ( %device $= "Voodoo2" )
            %this.add( "16", 0 );
      else
      {
            %resList = getResolutionList( %device );
            %resCount = getFieldCount( %resList );
      %count = 0;
            for ( %i = 0; %i < %resCount; %i++ )
            {
                  %bpp = getWord( getField( %resList, %i ), 2 );

                  // Only add to list if it isn't there already:
                  if ( %this.findText( %bpp ) == -1 )
         {
                        %this.add( %bpp, %count );
            %count++;
         }
            }
      }
}

function OptScreenshotMenu::init( %this )
{
   if( %this.findText("PNG") == -1 )
      %this.add("PNG", 0);
   if( %this.findText("JPEG") == - 1 )
      %this.add("JPEG", 1);
}

function optionsDlg::applyGraphics( %this )
{
      %newDriver = OptGraphicsDriverMenu.getText();
      %newRes = OptGraphicsResolutionMenu.getText();
      %newBpp = OptGraphicsBPPMenu.getText();
      %newFullScreen = OptGraphicsFullscreenToggle.getValue();
      $pref::Video::screenShotFormat = OptScreenshotMenu.getText();

      if ( %newDriver !$= $pref::Video::displayDevice )
      {
            setDisplayDevice( %newDriver, firstWord( %newRes ),
getWord( %newRes, 1 ), %newBpp, %newFullScreen );
            //OptionsDlg::deviceDependent( %this );
      }
      else
            setScreenMode( firstWord( %newRes ), getWord( %newRes, 1
), %newBpp, %newFullScreen );
}
```

8.2.9 Museum\client\ui\clientGui.cs

```
//--------------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
```

```
// Museum\client\ui\clientGui.cs
//--------------------------------------------------------------------------



new GameTSCtrl(PlayGui) {
   Profile = "GuiContentProfile";
   HorizSizing = "relative";
   VertSizing = "relative";
   position = "0 0";
   Extent = "640 480";
   MinExtent = "8 8";
   Visible = "1";
   Variable = "ap";
   Command = "ap";
   AltCommand = "ap";
   Accelerator = "ap";
   tooltip = "ap";
   langTableMod = "ap";
   applyFilterToChildren = "1";
   cameraZRot = "0";
   forceFOV = "0";
      helpTag = "0";
      noCursor = "1";

   new GuiTextCtrl(MuseumTag) {
      Profile = "GuiMediumTextProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "460 10";
      Extent = "260 60";
      MinExtent = "8 2";
      Visible = "1";
      textColor = "128 128 128 -2.22768e+038";
      text = "Modern   Art   Museum";
      maxLength = "255";
   };

new GuiTextCtrl(helpMessage) {
      Profile = "GuiTextProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "510 35";
      Extent = "260 60";
      MinExtent = "8 2";
      Visible = "0";
      textColor = "128 128 128 -2.22768e+038";
      text = "Push H for help";
      maxLength = "255";
   };

new GuiTextCtrl(PlaceTriggerControler) {
      Profile = "GuiMediumTextProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "460 30";
      Extent = "250 60";
      MinExtent = "8 2";
      Visible = "0";
      text = "Exhibition of sculptures";
      maxLength = "255";
```

```
   };


 new GuiTextEditCtrl(ArtistInfoTriggerControler) {
      Profile = "GuiTextEditProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "220 210";
      Extent = "260 120";
      MinExtent = "80 20";
      Visible = "0";
      text = " Artist's date of birth        #       Artist's
nationality " ;
      maxLength = "510";
   };
new GuiTextEditCtrl(ExhibitInfoTriggerControler) {
      Profile = "GuiTextEditProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "220 230";
      Extent = "260 120";
      MinExtent = "80 20";
      Visible = "0";
      text = "  Exhibit's type       #       Tecnique      #
Materials     #      Date" ;
      maxLength = "510";
   };

new GuiTextEditCtrl(ObjectTriggerControler) {
      Profile = "GuiTextEditProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "270 170";
      Extent = "150 120";
      MinExtent = "80 20";
      Visible = "0";
      text = "      Sculpture11      #        Artist11";
      maxLength = "510";
   };


 new GuiTextCtrl(TriggerControler) {
      Profile = "GuiBigTextProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "21 60";
      Extent = "140 50";
      MinExtent = "8 2";
      Visible = "0";
      text = "Entering trigger area";
      maxLength = "255";
   };


new GuiTextCtrl(TriggerControlerExit) {
      Profile = "GuiBigTextProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "21 60";
      Extent = "140 50";
      MinExtent = "8 2";
```

```
      Visible = "0";
      text = "Leaving trigger area";
      maxLength = "255";
   };

new GuiTextCtrl(ExpoTriggerControler) {
      Profile = "GuiBigTextProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "30 200";
      Extent = "200 50";
      MinExtent = "8 2";
      Visible = "0";
      text = "Ground Floor  #  Exhibition of sculptures";
      maxLength = "255";
   };



new GuiTextCtrl(InfoHouseTriggerControler) {
      Profile = "GuiBigTextProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "100 200";
      Extent = "200 50";
      MinExtent = "8 2";
      Visible = "0";
      text = "Welcome to the information office!";
      maxLength = "255";
   };




new GuiTextCtrl(StartTriggerControler1) {
      Profile = "GuiBigTextProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "15 120";
      Extent = "200 50";
      MinExtent = "8 2";
      Visible = "0";
      text = "Information Office ";
      maxLength = "255";
   };
new GuiTextCtrl(StartTriggerControler2) {
      Profile = "GuiBigTextProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "390 120";
      Extent = "200 50";
      MinExtent = "8 2";
      Visible = "0";
      text = "Modern art museum ";
      maxLength = "255";
   };


};
```

8.2.10 Museum\client\ui\loadingGui.cs

```
//--------------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\client\ui\loadingGui.cs
//--------------------------------------------------------------------------



new GuiControlProfile ("LoadingGuiContentProfile")
{
   opaque = true;
   fillColor = "200 200 200";
   border = true;
   borderColor   = "0 0 0";
};

new GuiChunkedBitmapCtrl(LoadingGui) {
   profile = "GuiContentProfile";
   horizSizing = "width";
   vertSizing = "height";
   position = "0 0";
   extent = "640 480";
   minExtent = "8 8";
   visible = "1";
   helpTag = "0";
   bitmap = "./background.jpg";
   useVariable = "0";
   tile = "0";

   new GuiControl() {
      profile = "LoadingGuiContentProfile";
      horizSizing = "center";
      vertSizing = "center";
      position = "30 110";
      extent = "455 308";
      minExtent = "8 8";
      visible = "1";
      helpTag = "0";

      new GuiTextCtrl(LOAD_MapName) {
         profile = "GuiMediumTextProfile";
         horizSizing = "right";
         vertSizing = "bottom";
         position = "7 6";
         extent = "100 28";
         minExtent = "8 8";
         visible = "1";
         helpTag = "0";
         text = "Map Name";
         maxLength = "255";
      };
      new GuiMLTextCtrl(LOAD_MapDescription) {
         profile = "GuiMLTextProfile";
         horizSizing = "right";
         vertSizing = "bottom";
         position = "7 62";
```

```
         extent = "303 16";
         minExtent = "8 8";
         visible = "1";
         helpTag = "0";
         lineSpacing = "2";
         allowColorChars = "0";
         maxChars = "-1";
      };
      new GuiProgressCtrl(LoadingProgress) {
         profile = "GuiProgressProfile";
         horizSizing = "right";
         vertSizing = "top";
         position = "128 262";
         extent = "262 25";
         minExtent = "8 8";
         visible = "1";
         helpTag = "0";

         new GuiTextCtrl(LoadingProgressTxt) {
            profile = "GuiProgressTextProfile";
            horizSizing = "right";
            vertSizing = "bottom";
            position = "-4 3";
            extent = "262 20";
            minExtent = "8 8";
            visible = "1";
            helpTag = "0";
            text = "LOADING MISSION";
            maxLength = "255";
         };
      };
      new GuiButtonCtrl() {
         profile = "GuiButtonProfile";
         horizSizing = "right";
         vertSizing = "top";
         position = "58 262";
         extent = "65 25";
         minExtent = "20 20";
         visible = "1";
         command = "disconnect();";
         accelerator = "escape";
         helpTag = "0";
         text = "CANCEL";
      };
   };
};
```

8.2.11 Museum\client\ui\mainMenuGui.cs

```
//--------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\client\ui\mainMenuGui.cs
//--------------------------------------------------------------------


new GuiChunkedBitmapCtrl(MainMenuGui) {
   Profile = "GuiContentProfile";
```

```
HorizSizing = "width";
VertSizing = "height";
position = "0 0";
Extent = "640 480";
MinExtent = "8 8";
Visible = "1";
bitmap = "./colours";
useVariable = "0";
tile = "0";
   helpTag = "0";

new GuiChunkedBitmapCtrl() {
   Profile = "GuiDefaultProfile";
   HorizSizing = "relative";
   VertSizing = "relative";
   position = "520 384";
   Extent = "92 72";
   MinExtent = "8 2";
   Visible = "1";
   useVariable = "0";
   tile = "0";

   new GuiBitmapButtonCtrl() {
      Profile = "GuiBorderButtonProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "0 0";
      Extent = "92 72";
      MinExtent = "8 2";
      Visible = "1";
      Command = "quit();";
      tooltipprofile = "GuiToolTipProfile";
      tooltip = "Close down the engine";
      text = "Button";
      groupNum = "-1";
      buttonType = "PushButton";
      bitmap = "./buttons/exits";
   };
};
new GuiChunkedBitmapCtrl() {
   Profile = "GuiDefaultProfile";
   HorizSizing = "relative";
   VertSizing = "relative";
   position = "28 192";
   Extent = "200 148";
   MinExtent = "8 2";
   Visible = "1";
   bitmap = "./gray_bar";
   useVariable = "0";
   tile = "0";

   new GuiBitmapButtonCtrl() {
      Profile = "GuiBorderButtonProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "-4 -16";
      Extent = "208 174";
      MinExtent = "8 2";
      Visible = "1";
      Command = "loadMyMission();";
      tooltipprofile = "GuiToolTipProfile";
```

```
         tooltip = "Starting the visit to the museum";
         text = "Button";
         groupNum = "-1";
         buttonType = "PushButton";
         bitmap = "./buttons/start";
      };
   };
   new GuiChunkedBitmapCtrl() {
      Profile = "GuiDefaultProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "520 32";
      Extent = "88 69";
      MinExtent = "8 2";
      Visible = "1";
      bitmap = "./gray_bar";
      useVariable = "0";
      tile = "0";

      new GuiBitmapButtonCtrl() {
         Profile = "GuiBorderButtonProfile";
         HorizSizing = "relative";
         VertSizing = "relative";
         position = "0 0";
         Extent = "88 69";
         MinExtent = "8 2";
         Visible = "1";
         Command = "GuiEdit();";
         tooltipprofile = "GuiToolTipProfile";
         tooltip = "Open the GUI Editor, which helps you create
graphical user interfaces for your games";
         text = "Button";
         groupNum = "-1";
         buttonType = "PushButton";
         bitmap = "./buttons/guieditor";
      };
   };
   new GuiChunkedBitmapCtrl() {
      Profile = "GuiDefaultProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "560 12";
      Extent = "80 452";
      MinExtent = "8 2";
      Visible = "0";
      bitmap = "./gray_bar";
      useVariable = "0";
      tile = "0";
   };
   new GuiChunkedBitmapCtrl() {
      Profile = "GuiDefaultProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "324 204";
      Extent = "52 112";
      MinExtent = "8 2";
      Visible = "0";
      bitmap = "./gray_bar";
      useVariable = "0";
      tile = "0";
```

```
      new GuiBitmapButtonCtrl() {
         Profile = "GuiBorderButtonProfile";
         HorizSizing = "relative";
         VertSizing = "relative";
         position = "0 312";
         Extent = "8 40";
         MinExtent = "8 2";
         Visible = "0";
         Command =
"gotoWebpage(\"http://www.garagegames.com/mg/forums/result.area.php?q
a=10\");";
         tooltipprofile = "GuiToolTipProfile";
         tooltip = "Talk to fellow Torque developers on the private
Torque owner\'s forum";
         text = "Button";
         groupNum = "-1";
         buttonType = "PushButton";
         bitmap = "./buttons/forum";
      };
   };
   new GuiChunkedBitmapCtrl() {
      Profile = "GuiDefaultProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "520 300";
      Extent = "92 68";
      MinExtent = "8 2";
      Visible = "1";
      bitmap = "./gray_bar";
      useVariable = "0";
      tile = "0";

      new GuiBitmapButtonCtrl() {
         Profile = "GuiBorderButtonProfile";
         HorizSizing = "relative";
         VertSizing = "relative";
         position = "0 -9";
         Extent = "95 81";
         MinExtent = "8 2";
         Visible = "1";
         Command = "Canvas.pushDialog(optionsDlg);";
         tooltipprofile = "GuiToolTipProfile";
         tooltip = "Adjust basic video and other options";
         text = "Button";
         groupNum = "-1";
         buttonType = "PushButton";
         bitmap = "./buttons/option";
      };
   };
   new GuiChunkedBitmapCtrl() {
      Profile = "GuiDefaultProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "517 120";
      Extent = "95 68";
      MinExtent = "8 2";
      Visible = "1";
      bitmap = "./gray_bar";
      useVariable = "0";
      tile = "0";
```

```
      new GuiBitmapButtonCtrl() {
         Profile = "GuiBorderButtonProfile";
         HorizSizing = "relative";
         VertSizing = "relative";
         position = "0 0";
         Extent = "95 68";
         MinExtent = "8 2";
         Visible = "1";
         Command = "toggleEditor(1);";
         tooltipprofile = "GuiToolTipProfile";
         tooltip = "Open the World Editor, which provides tools for
creating and editing your game worlds";
         text = "Button";
         groupNum = "-1";
         buttonType = "PushButton";
         bitmap = "./buttons/worldeditor";
      };
   };
   new GuiChunkedBitmapCtrl() {
      Profile = "GuiDefaultProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "272 204";
      Extent = "48 112";
      MinExtent = "8 2";
      Visible = "0";
      bitmap = "./gray_bar";
      useVariable = "0";
      tile = "0";

      new GuiBitmapButtonCtrl() {
         Profile = "GuiBorderButtonProfile";
         HorizSizing = "right";
         VertSizing = "top";
         position = "495 382";
         Extent = "32 44";
         MinExtent = "8 2";
         Visible = "0";
         Command =
"gotoWebpage(\"http://www.garagegames.com/index.php?sec=mg&mod=resour
ce&page=result&qrt=News&qrf=tska&qsf=posted&qsd=desc\");";
         tooltipprofile = "GuiToolTipProfile";
         tooltip = "Get the latest Torque news";
         text = "Button";
         groupNum = "-1";
         buttonType = "PushButton";
         bitmap = "./buttons/news";
      };
   };
   new GuiChunkedBitmapCtrl() {
      Profile = "GuiDefaultProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "304 204";
      Extent = "80 112";
      MinExtent = "8 2";
      Visible = "0";
      bitmap = "./gray_bar";
      useVariable = "0";
      tile = "0";
```

```
      new GuiBitmapButtonCtrl() {
         Profile = "GuiBorderButtonProfile";
         HorizSizing = "right";
         VertSizing = "top";
         position = "501 446";
         Extent = "26 31";
         MinExtent = "8 2";
         Visible = "0";
         Command =
"gotoWebpage(\"http://www.garagegames.com/mg/projects/tge/\");";
         tooltipprofile = "GuiToolTipProfile";
         tooltip = "See the Torque Tutorials homepage and learn more
about how to use the engine";
         text = "Button";
         groupNum = "-1";
         buttonType = "PushButton";
         bitmap = "./buttons/help";
      };
   };
   new GuiChunkedBitmapCtrl() {
      Profile = "GuiDefaultProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "288 204";
      Extent = "80 112";
      MinExtent = "8 2";
      Visible = "0";
      bitmap = "./gray_bar";
      useVariable = "0";
      tile = "0";

      new GuiBitmapButtonCtrl() {
         Profile = "GuiBorderButtonProfile";
         HorizSizing = "right";
         VertSizing = "top";
         position = "495 557";
         Extent = "35 42";
         MinExtent = "8 2";
         Visible = "0";
         Command = "gotoWebpage(\"http://tdn.garagegames.com/\");";
         tooltipprofile = "GuiToolTipProfile";
         tooltip = "Go to the Torque Developer Network, a community-
oriented site for Torque documentation";
         text = "Button";
         groupNum = "-1";
         buttonType = "PushButton";
         bitmap = "./buttons/tdn";
      };
   };
   new GuiChunkedBitmapCtrl() {
      Profile = "GuiDefaultProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "520 208";
      Extent = "92 68";
      MinExtent = "8 2";
      Visible = "1";
      bitmap = "./gray_bar";
      useVariable = "0";
      tile = "0";
```

```
      new GuiBitmapButtonCtrl() {
         Profile = "GuiBorderButtonProfile";
         HorizSizing = "relative";
         VertSizing = "relative";
         position = "-2 -10";
         Extent = "100 84";
         MinExtent = "8 2";
         Visible = "1";
         Command = "ToggleConsole(1);";
         tooltipprofile = "GuiToolTipProfile";
         tooltip = "View the scripting Console, which allows you to
enter TorqueScript commands on the fly";
         text = "Button";
         groupNum = "-1";
         buttonType = "PushButton";
         bitmap = "./buttons/consol";
      };
   };
   new GuiChunkedBitmapCtrl() {
      Profile = "GuiDefaultProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "280 204";
      Extent = "72 112";
      MinExtent = "8 2";
      Visible = "0";
      bitmap = "./gray_bar";
      useVariable = "0";
      tile = "0";

      new GuiBitmapButtonCtrl() {
         Profile = "GuiBorderButtonProfile";
         HorizSizing = "right";
         VertSizing = "top";
         position = "398 610";
         Extent = "46 42";
         MinExtent = "8 2";
         Visible = "0";
         Command = "MessageBoxOK(\"Tutorial\", \"Note: In order to
get to the tutorial right now, you need to go to your SDK install
directory, then open the example folder and double-click
\'GettingStarted.pdf\'.\");";
         tooltipprofile = "GuiToolTipProfile";
         tooltip = "Read the Basic Getting Started Tutorial and learn
the first steps of using some of Torque\'s tools";
         text = "Button";
         groupNum = "-1";
         buttonType = "PushButton";
         bitmap = "./buttons/tutorial";
      };
   };
   new GuiMLTextCtrl(RSSFeedMLText) {
      Profile = "GuiMLTextProfile";
      HorizSizing = "width";
      VertSizing = "top";
      position = "4 390";
      Extent = "326 75";
      MinExtent = "8 2";
      Visible = "0";
      lineSpacing = "2";
      allowColorChars = "0";
```

```
      maxChars = "-1";
   };
   new GuiButtonCtrl() {
      Profile = "GuiButtonProfile";
      HorizSizing = "relative";
      VertSizing = "relative";
      position = "268 376";
      Extent = "140 32";
      MinExtent = "8 2";
      Visible = "0";
      Command = "loadMyMission();";
      text = "S T A R T";
      groupNum = "-1";
      buttonType = "PushButton";
   };
};
//--------------------------------------------------------------------
// RSS ticker configuration:
$RSSFeed::serverName = "feeds.feedburner.com";
$RSSFeed::serverPort = 80;
$RSSFeed::serverURL  = "/garagegames/rss/product/tge/oobe";
$RSSFeed::userAgent = "Torque/1.4";

function RSSFeedObject::onConnected(%this)
{
      echo("RSS Feed");
      echo("   - Requesting RSS data at URL: " @ $RSSFeed::serverURL
);

      // Reset some useful state information.
      $RSSFeed::lineCount = 0;
      $RSSFeed::requestResults = "";

      // Request our RSS.
      %this.send("GET " @ $RSSFeed::serverURL @ " HTTP/1.0\nHost: " @
$RSSFeed::serverName @ "\nUser-Agent: " @ $RSSFeed::userAgent @
"\n\r\n\r\n");
}

function RSSFeedObject::onLine(%this, %line)
{
      // Collate info.
      $RSSFeed::lineCount++;
      $RSSFeed::requestResults = $RSSFeed::requestResults @ %line;
}

function RSSFeedObject::getTagContents(%this, %string, %tag,
%startChar)
{
 // This function occasionally makes Torque hard crash. It doesn't
 // seem to do it anymore but be careful!

      // Ok, get thing between <%tag> and </%tag> after char #
      // %startChar in the passed string.

      %startTag = "<" @ %tag @ ">";
      %endTag   = "</" @ %tag @ ">";

      %startTagOffset = strpos(%string, %startTag, %startChar);

      // Compensate for presence of start tag.
```

```
      %startOffset = %startTagOffset + strlen(%startTag);

      // Ok, now look for end tag.
      %endTagOffset = strpos(%string, %endTag, %startOffset - 1);

      // If we didn't find it, bail.
      if(%endTagOffset < 0)
            return "";

      // Evil hack - store last found item in a global.
       %this.lastOffset = %endTagOffset;

      // And get & return the substring between the tags.
      %result = getSubStr(%string, %startOffset, %endTagOffset -
%startOffset);

      // Do a little mojo to deal with &quot; and some other
htmlentities.
      %result = strreplace(%result, "&quot;", "\"");
      %result = strreplace(%result, "&amp;",  "&");

      return %result;
}

function RSSFeedObject::onDisconnect(%this)
{
      // Ok, we have a full buffer now, hopefully. Let's process it.
      echo("   - Got " @ $RSSFeed::lineCount @ " lines.");

      // We want the feed title and the first three headlines +
links.

      // Feed title - get the first <title> occurence in the string.
      %title = %this.getTagContents($RSSFeed::requestResults,
"title", 0);
      %titleLink = %this.getTagContents($RSSFeed::requestResults,
"link", 0);

      echo("   - Feed title: '" @ %title @ "'");
      echo("   - Feed link:  '" @ %titleLink @ "'");

      // Ok, get the first three headlines, if any...
      for(%i = 0; %i<3; %i++)
      {
         %headline[%i]     =
%this.getTagContents($RSSFeed::requestResults, "title",
%this.lastOffset);
         %headlineLink[%i] =
%this.getTagContents($RSSFeed::requestResults, "link",
%this.lastOffset);

         // Skip the content - it's not going to do anything but
confuse us.
         %garbage          =
%this.getTagContents($RSSFeed::requestResults, "content:encoded",
%this.lastOffset);

    //  And debug spam...
    echo("   - Headline      #" @ %i @ " : '" @ %headline[%i] @ "'");
    echo("   - Headline Link #" @ %i @ " : '" @ %headlineLink[%i] @
"'");
```

```
      }

  // Generate contents for our ML control.
  RSSFeedMLText.setText("<lmargin%:2><font:Arial Bold:20>" @ %title @
"<font:Arial:14>\n");
  for(%i=0; %i<3; %i++)
     RSSFeedMLText.addText("<a:" @ getSubstr(%headlineLink[%i], 7,
1000) @ ">" @ %headline[%i] @ "</a>\n", false);
  RSSFeedMLText.addText("<just:right><a:" @ getSubstr(%titleLink, 7,
1000) @ ">" @ "[ Read more... ]" @ "</a>", true);

}

function kickOffRSS()
{
   new TCPObject(RSSFeedObject);
   RSSFeedObject.connect($RSSFeed::serverName @ ":" @
$RSSFeed::serverPort);
}

function MainMenuGui::onWake(%this)
{
   // Kick off an update on next tick.
   if(!$pref::RSS::disableFeedCheck)
      schedule(50, 0, kickOffRSS);
}
//--------------------------------------------------------------------------------
```

8.2.12 Museum\client\ui\optionsDlg.cs

```
//--------------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\client\ui\optionsDlg.cs
//--------------------------------------------------------------------------

new GuiControl(optionsDlg) {
   Profile = "GuiDefaultProfile";
   HorizSizing = "right";
   VertSizing = "bottom";
   position = "0 0";
   Extent = "800 600";
   MinExtent = "8 8";
   Visible = "1";
      helpTag = "0";

   new GuiWindowCtrl() {
      Profile = "GuiWindowProfile";
      HorizSizing = "center";
      VertSizing = "center";
      position = "211 148";
      Extent = "377 303";
      MinExtent = "8 8";
      Visible = "1";
      text = "Options";
      maxLength = "255";
      resizeWidth = "0";
```

```
resizeHeight = "0";
canMove = "1";
canClose = "1";
canMinimize = "0";
canMaximize = "0";
MinSize = "50 50";
closeCommand = "Canvas.popDialog(optionsDlg);";
   helpTag = "0";

new GuiButtonCtrl(OK_Button) {
   Profile = "GuiButtonProfile";
   HorizSizing = "right";
   VertSizing = "bottom";
   position = "305 270";
   Extent = "60 23";
   MinExtent = "8 8";
   Visible = "1";
   Command = "Canvas.popDialog(optionsDlg);";
   text = "OK";
   groupNum = "-1";
   buttonType = "PushButton";
      helpTag = "0";
};
new GuiControl(OptGraphicsPane) {
   Profile = "GuiWindowProfile";
   HorizSizing = "right";
   VertSizing = "bottom";
   position = "9 55";
   Extent = "357 208";
   MinExtent = "8 8";
   Visible = "1";
      helpTag = "0";

   new GuiTextCtrl() {
      Profile = "GuiTextProfile";
      HorizSizing = "right";
      VertSizing = "bottom";
      position = "21 10";
      Extent = "70 18";
      MinExtent = "8 8";
      Visible = "1";
      text = "Display Driver:";
      maxLength = "255";
         helpTag = "0";
   };
   new GuiTextCtrl() {
      Profile = "GuiTextProfile";
      HorizSizing = "right";
      VertSizing = "bottom";
      position = "21 34";
      Extent = "53 18";
      MinExtent = "8 8";
      Visible = "1";
      text = "Resolution:";
      maxLength = "255";
         helpTag = "0";
   };
   new GuiCheckBoxCtrl(OptGraphicsFullscreenToggle) {
      Profile = "GuiCheckBoxProfile";
      HorizSizing = "right";
      VertSizing = "bottom";
```

```
   position = "21 120";
   Extent = "137 25";
   MinExtent = "8 8";
   Visible = "1";
   Variable = "$pref::Video::fullScreen";
   text = "Fullscreen Video";
   groupNum = "-1";
   buttonType = "ToggleButton";
      maxLength = "255";
      helpTag = "0";
};
new GuiButtonCtrl() {
   Profile = "GuiButtonProfile";
   HorizSizing = "right";
   VertSizing = "bottom";
   position = "149 171";
   Extent = "78 23";
   MinExtent = "8 8";
   Visible = "1";
   Command = "optionsDlg.applyGraphics();";
   text = "Apply";
   groupNum = "-1";
   buttonType = "PushButton";
      helpTag = "0";
};
new GuiPopUpMenuCtrl(OptGraphicsDriverMenu) {
   Profile = "GuiPopUpMenuProfile";
   HorizSizing = "right";
   VertSizing = "bottom";
   position = "113 10";
   Extent = "130 23";
   MinExtent = "8 8";
   Visible = "1";
   maxLength = "255";
   maxPopupHeight = "200";
      helpTag = "0";
};
new GuiPopUpMenuCtrl(OptGraphicsResolutionMenu) {
   Profile = "GuiPopUpMenuProfile";
   HorizSizing = "right";
   VertSizing = "bottom";
   position = "113 36";
   Extent = "130 23";
   MinExtent = "8 8";
   Visible = "1";
   maxLength = "255";
   maxPopupHeight = "200";
      helpTag = "0";
};
new GuiTextCtrl() {
   Profile = "GuiTextProfile";
   HorizSizing = "right";
   VertSizing = "bottom";
   position = "21 60";
   Extent = "46 18";
   MinExtent = "8 8";
   Visible = "1";
   text = "Bit Depth:";
   maxLength = "255";
      helpTag = "0";
};
```

```
         new GuiPopUpMenuCtrl(OptGraphicsBPPMenu) {
            Profile = "GuiPopUpMenuProfile";
            HorizSizing = "right";
            VertSizing = "bottom";
            position = "113 62";
            Extent = "130 23";
            MinExtent = "8 8";
            Visible = "1";
            maxLength = "255";
            maxPopupHeight = "200";
               helpTag = "0";
         };
         new GuiTextCtrl() {
            Profile = "GuiTextProfile";
            HorizSizing = "right";
            VertSizing = "bottom";
            position = "21 86";
            Extent = "59 18";
            MinExtent = "8 2";
            Visible = "1";
            text = "Screenshot:";
            maxLength = "255";
               helpTag = "0";
         };
         new GuiPopUpMenuCtrl(OptScreenshotMenu) {
            Profile = "GuiPopUpMenuProfile";
            HorizSizing = "right";
            VertSizing = "bottom";
            position = "113 88";
            Extent = "130 23";
            MinExtent = "8 2";
            Visible = "1";
            maxLength = "255";
            maxPopupHeight = "200";
               helpTag = "0";
         };

      };
   };
};
//--------------------------------------------------------------------------
```

8.3 Δεδομένα (data).

8.3.1 Museum\data\missions\MuseumMission.mis

```
//--------------------------------------------------------------------------
// Mordern Art Museum
// Created by Antonopoulou C.
// Museum\data\missions\MuseumMission.mis
//--------------------------------------------------------------------------



new SimGroup(MissionGroup) {

   new StaticShape(guide1) {
      position = "42 -312 3.6";
```

```
   rotation = "0 0 1 180";
   scale = "1 1 1";
   dataBlock = "Guide";
};
new ScriptObject(MissionInfo) {
      desc0 = "Created by Antonopoulou C.";
      name = "Modern Art Museum";
};
new Sky(Sky) {
   position = "336 136 0";
   rotation = "1 0 0 0";
   scale = "1 1 1";
   materialList = "~/data/skies/sky_day.dml";
   cloudHeightPer[0] = "0";
   cloudHeightPer[1] = "0.2";
   cloudHeightPer[2] = "0.1";
   cloudSpeed1 = "0.0004";
   cloudSpeed2 = "0.0006";
   cloudSpeed3 = "0.0008";
   visibleDistance = "500";
   fogDistance = "500";
   fogColor = "0.82 0.828 0.844 1";
   fogStorm1 = "0";
   fogStorm2 = "0";
   fogStorm3 = "0";
   fogVolume1 = "300 0 20";
   fogVolume2 = "0 0 0";
   fogVolume3 = "0 0 0";
   fogVolumeColor1 = "128 128 128 -2.22768e+038";
   fogVolumeColor2 = "128 128 128 0";
   fogVolumeColor3 = "128 128 128 -1.70699e+038";
   windVelocity = "1 1 0";
   windEffectPrecipitation = "1";
   SkySolidColor = "0.547 0.641 0.789 0";
   useSkyTextures = "1";
   renderBottomTexture = "0";
   noRenderBans = "0";
      locked = "true";
};
new SimGroup(PlayerDropPoints) {

   new SpawnSphere(ss1) {
      position = "-34 -372 9";
      rotation = "0 0 1 89.3825";
      scale = "1 1 2";
      dataBlock = "SpawnSphereMarker";
      radius = "100";
      sphereWeight = "100";
      indoorWeight = "100";
      outdoorWeight = "100";
         lockCount = "0";
         locked = "False";
         homingCount = "0";
   };
};
new SimGroup(Triggers) {

   new Trigger(Object15Trigger) {
      position = "70.7358 -434.201 0.5";
      rotation = "1 0 0 0";
      scale = "2 2 2";
```

```
         dataBlock = "Object15Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(DownFloorTrigger) {
         position = "55.5 -402 -3.6";
         rotation = "1 0 0 0";
         scale = "39 43 7";
         dataBlock = "ExpoOneTrigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(Object23Trigger) {
         position = "84.8654 -430.628 12.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object23Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(Object22Trigger) {
         position = "84.3763 -412.681 12.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object22Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(Object24Trigger) {
         position = "84.763 -441.618 12.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object24Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(Object25Trigger) {
         position = "61.7864 -446.604 12.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object25Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(Object26Trigger) {
         position = "57.8462 -430.65 12.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object26Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(Object27Trigger) {
         position = "57.4474 -415.77 12.5";
```

```
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object27Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(Object21Trigger) {
         position = "65.0208 -398.142 12.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object21Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(Object28Trigger) {
         position = "57.5595 -403.367 12.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object28Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(Obect12Trigger) {
         position = "72.3911 -411.62 0.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object12Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(Object11Trigger) {
         position = "64.7484 -411.35 0.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object11Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(Obect13Trigger) {
         position = "78.7018 -412.127 0.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object13Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(Obect14Trigger) {
         position = "80.669 -433.924 0.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object14Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
```

```
      new Trigger(Obect16Trigger) {
         position = "64.2668 -433.898 0.5";
         rotation = "1 0 0 0";
         scale = "2 2 2";
         dataBlock = "Object16Trigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(infoOfficeTrigger) {
         position = "32.5 -309 2";
         rotation = "1 0 0 0";
         scale = "16 16 6";
         dataBlock = "InfoHouseTrigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(StartTrigger) {
         position = "-34.5 -370 8";
         rotation = "0 0 -1 0.0395647";
         scale = "1 5 1";
         dataBlock = "StartTrigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
      new Trigger(UpFloorTrigger) {
         position = "55 -396 9.6";
         rotation = "1 0 0 0";
         scale = "33 55 11";
         dataBlock = "ExpoTwoTrigger";
         polyhedron = "0.0000000 0.0000000 0.0000000 1.0000000
0.0000000 0.0000000 0.0000000 -1.0000000 0.0000000 0.0000000
0.0000000 1.0000000";
      };
   };
   new SimGroup(sculpture_BasesBars) {

      new InteriorInstance(scalebase3) {
         position = "85.0888 -411.736 -8.4";
         rotation = "1 0 0 180";
         scale = "0.6 0.6 0.6";
         interiorFile = "~/data/interiors/scale1min.dif";
         useGLLighting = "0";
         showTerrainInside = "0";
      };
      new InteriorInstance(scalebase2) {
         position = "79.7208 -411 -9.8727";
         rotation = "1 0 0 180";
         scale = "0.7 0.7 0.7";
         interiorFile = "~/data/interiors/scale1min.dif";
         useGLLighting = "0";
         showTerrainInside = "0";
      };
      new StaticShape(wbar13) {
         position = "78 -414.72 1.55";
         rotation = "1 0 0 0";
         scale = "3 0.1 0.1";
         dataBlock = "Base13Item";
      };
```

```
new StaticShape(wbar14) {
   position = "80.2 -433.813 1.55";
   rotation = "1 0 0 0";
   scale = "3 0.1 0.1";
   dataBlock = "Base14Item";
};
new StaticShape(wbar15) {
   position = "70.8 -433.813 1.55";
   rotation = "1 0 0 0";
   scale = "3 0.1 0.1";
   dataBlock = "Base15Item";
};
new StaticShape(wbar16) {
   position = "63.5 -433.813 1.55";
   rotation = "1 0 0 0";
   scale = "3 0.1 0.1";
   dataBlock = "Base16Item";
};
new StaticShape(wbar11) {
   position = "64.5 -414.72 1.55";
   rotation = "1 0 0 0";
   scale = "3 0.1 0.1";
   dataBlock = "Base11Item";
};
new StaticShape(wbar12) {
   position = "71.5 -414.72 1.55";
   rotation = "1 0 0 0";
   scale = "3 0.1 0.1";
   dataBlock = "Base12Item";
};
new InteriorInstance(scalebase1) {
   position = "57.301 -411 -8.43928";
   rotation = "0 1 0 179.909";
   scale = "0.6 0.6 0.6";
   interiorFile = "~/data/interiors/scale1min.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(scalebase4) {
   position = "91.9742 -436.656 -10";
   rotation = "1 0 0 180";
   scale = "1 0.7 0.7";
   interiorFile = "~/data/interiors/scale1min.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(scalebase5) {
   position = "75 -437.5 -10";
   rotation = "1 0 0 180";
   scale = "1 1 0.7";
   interiorFile = "~/data/interiors/scale1min.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(scalebase6) {
   position = "82.6144 -437 -9.8484";
   rotation = "1 0 0 180";
   scale = "1 0.7 0.7";
   interiorFile = "~/data/interiors/scale1min.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
```

```
   };
   new InteriorInstance(sculpturebar5) {
      position = "115.7 -449 2.3";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/objectbar.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(sculpturebar6) {
      position = "108.5 -449 2.3";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/objectbar.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(sculpturebar2) {
      position = "116.7 -429.865 2.3";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/objectbar.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(sculpturebar1) {
      position = "109.5 -429.865 2.3";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/objectbar.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(sculpturebar4) {
      position = "125.19 -449 2.3";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/objectbar.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(sculpturebar3) {
      position = "123.182 -429.865 2.3";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/objectbar.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
};
new SimGroup(sculptures) {

   new TSStatic() {
      position = "78.0764 -412.2 3";
      rotation = "1 0 0 0";
      scale = "1.5 1.5 1.5";
      shapeName = "~/data/shapes/scc.dts";
   };
   new TSStatic() {
      position = "63.5 -436.4 3.2";
      rotation = "-1 0 0 90";
```

```
      scale = "2.5 2.5 2.5";
      shapeName = "~/data/shapes/s4.dts";
   };
   new TSStatic(sculpture3) {
      position = "78.085 -412 3.60019";
      rotation = "0.00240878 0.707112 0.707098 179.659";
      scale = "0.6 0.6 0.7";
      shapeName = "~/data/shapes/aro.dts";
   };
   new InteriorInstance(sculpture4) {
      position = "-80.54 436.759 3.5";
      rotation = "0.0111104 0 0.999938 180";
      scale = "0.5 0.5 0.5";
      interiorFile = "~/data/interiors/sculpture4.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(sculpture4) {
      position = "80.5 -436.7 3.4";
      rotation = "1 0 0 180";
      scale = "0.5 0.5 0.5";
      interiorFile = "~/data/interiors/sculpture4.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(sculpture6) {
      position = "64.6077 -439.055 1.8";
      rotation = "1 0 0 0";
      scale = "0.7 1 0.7";
      interiorFile = "~/data/interiors/sculpture6.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(sculpture5) {
      position = "71.7668 -437.198 1.8";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/sculpture5.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(sculpture2) {
      position = "71.671 -411.2 2.80625";
      rotation = "1 0 0 0";
      scale = "0.5 0.5 0.5";
      interiorFile = "~/data/interiors/sculpture2.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(sculpture1) {
      position = "65.3232 -411.2 3.05937";
      rotation = "1 0 0 0";
      scale = "0.3 0.3 0.3";
      interiorFile = "~/data/interiors/sculpture1.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
};
new SimGroup(pictureBars) {

   new StaticShape(wbar25) {
```

```
   position = "64.3 -446.61 13.55";
   rotation = "1 0 0 0";
   scale = "12 0.05 0.05";
   dataBlock = "Base25Item";
};
new InteriorInstance(pbar5) {
   position = "199.2 -461.823 14.3";
   rotation = "1 0 0 0";
   scale = "3 1 1";
   interiorFile = "~/data/interiors/objectbar.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(pbar8) {
   position = "104.5 -428.5 14.2";
   rotation = "1 0 0 0";
   scale = "1 1.5 1";
   interiorFile = "~/data/interiors/objectbar1.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new StaticShape(wbar28) {
   position = "59.54 -405.953 13.45";
   rotation = "1 0 0 0";
   scale = "0.1 4 0.1";
   dataBlock = "Base28Item";
};
new InteriorInstance(pbar7) {
   position = "104.5 -442 14.2";
   rotation = "1 0 0 0";
   scale = "1 1.5 1";
   interiorFile = "~/data/interiors/objectbar1.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new StaticShape(wbar27) {
   position = "59.54 -419.473 13.44";
   rotation = "1 0 0 0";
   scale = "0.1 4 0.1";
   dataBlock = "Base27Item";
};
new InteriorInstance(pbar6) {
   position = "104.8 -472 14.1";
   rotation = "1 0 0 0";
   scale = "1 2.5 1";
   interiorFile = "~/data/interiors/objectbar1.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new StaticShape(wbar26) {
   position = "59.835 -434.5 13.35";
   rotation = "1 0 0 0";
   scale = "0.1 10 0.1";
   dataBlock = "Base26Item";
};
new InteriorInstance(bar5a) {
   position = "213.4 -461.823 14.4219";
   rotation = "1 0 0 0";
   scale = "3 1 1";
   interiorFile = "~/data/interiors/objectbar.dif";
   useGLLighting = "0";
```

```
   showTerrainInside = "0";
};
new InteriorInstance(bar5b) {
   position = "139 -461.823 14.4219";
   rotation = "1 0 0 0";
   scale = "1.5 1 1";
   interiorFile = "~/data/interiors/objectbar.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new StaticShape(wbar25a) {
   position = "78.5 -446.63 13.68";
   rotation = "1 0 0 0";
   scale = "12 0.1 0.1";
   dataBlock = "Base25Item";
};
new StaticShape(wbar25b) {
   position = "71.5 -446.63 13.68";
   rotation = "1 0 0 0";
   scale = "6 0.1 0.1";
   dataBlock = "Base25Item";
};
new InteriorInstance(pbar4) {
   position = "129.5 -463.9 14.5";
   rotation = "1 0 0 0";
   scale = "1 1.5 1.5";
   interiorFile = "~/data/interiors/objectbar1.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(pbar3) {
   position = "129.498 -474.009 14.5";
   rotation = "1 0 0 0";
   scale = "1 3 1.5";
   interiorFile = "~/data/interiors/objectbar1.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(pbar2) {
   position = "129.5 -456.2 14.5";
   rotation = "1 0 0 0";
   scale = "1 3 1.5";
   interiorFile = "~/data/interiors/objectbar1.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new StaticShape(wbar24) {
   position = "84.5 -441.148 13.3659";
   rotation = "1 0 0 0";
   scale = "0.1 5 0.1";
   dataBlock = "Base24Item";
};
new StaticShape(wbar22) {
   position = "84.5 -411.2 13.4";
   rotation = "1 0 0 0";
   scale = "0.1 12 0.1";
   dataBlock = "Base22Item";
};
new StaticShape(wbar23) {
   position = "84.5 -429 13.4";
   rotation = "1 0 0 0";
```

```
      scale = "0.1 12 0.1";
      dataBlock = "Base23Item";
   };
   new InteriorInstance(pbar1a) {
      position = "213 -415 14.3";
      rotation = "1 0 0 0";
      scale = "3 1 1";
      interiorFile = "~/data/interiors/objectbar.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(pbar1) {
      position = "204 -415 14.3";
      rotation = "1 0 0 0";
      scale = "3 1 1";
      interiorFile = "~/data/interiors/objectbar.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new StaticShape(wbar21) {
      position = "69.1 -399.86 13.56";
      rotation = "1 0 0 0";
      scale = "12 0.1 0.1";
      dataBlock = "Base21Item";
   };
   new StaticShape(wbar21a) {
      position = "78.1 -399.86 13.56";
      rotation = "1 0 0 0";
      scale = "12 0.1 0.1";
      dataBlock = "Base21Item";
   };
};
new SimGroup(pictures) {

   new InteriorInstance(picture5) {
      position = "97.6295 -451 17.35";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/picture.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(picture6) {
      position = "80.3426 -435.215 17";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/picture1.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(picture2) {
      position = "122.626 -410.48 17.0194";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/picture2.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(picture3) {
      position = "121.607 -428.311 16.5925";
      rotation = "1 0 0 0";
```

```
      scale = "1 1 1";
      interiorFile = "~/data/interiors/picture3.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(picture4) {
      position = "124 -441.368 16.8548";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/picture4.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(picture8) {
      position = "81.3697 -406.144 16.6";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/picture5.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(picture1) {
      position = "103.946 -396.66 17.4313";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/picture6.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(picture7) {
      position = "81.5499 -419.74 20.2";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/picture7.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
};
new SimGroup(pillars) {

   new InteriorInstance(pillarUpUpRight) {
      position = "77.6 -438 5.2";
      rotation = "1 0 0 0";
      scale = "1.3 1.3 1.8";
      interiorFile = "~/data/interiors/pillar.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(pillarUpRR) {
      position = "77.6 -438.9 6";
      rotation = "1 0 0 90";
      scale = "1.3 1.3 2.4";
      interiorFile = "~/data/interiors/pillar.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(pillarUpRRR) {
      position = "77.6 -446.3 6";
      rotation = "1 0 0 90";
      scale = "1.3 1.3 2.4";
      interiorFile = "~/data/interiors/pillar.dif";
```

```
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(pillarUpLL) {
   position = "77.6 -418 6";
   rotation = "1 0 0 90";
   scale = "1.3 1.3 2.4";
   interiorFile = "~/data/interiors/pillar.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(pillarUpLLL) {
   position = "77.6 -410.3 6";
   rotation = "1 0 0 90";
   scale = "1.3 1.3 2.4";
   interiorFile = "~/data/interiors/pillar.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(pillarDownRight) {
   position = "77.6 -431.5 0";
   rotation = "1 0 0 90";
   scale = "1.3 0.8 2.2";
   interiorFile = "~/data/interiors/pillar.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(pillarDownLeft) {
   position = "77.6 -424.9 0";
   rotation = "1 0 0 90";
   scale = "1.3 0.8 2.2";
   interiorFile = "~/data/interiors/pillar.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(pillarUpUpLeft) {
   position = "77.6 -409 5.2";
   rotation = "1 0 0 0";
   scale = "1.3 1.3 1.8";
   interiorFile = "~/data/interiors/pillar.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(pillarLeft) {
   position = "77.6 -416.9 -0.9";
   rotation = "1 0 0 0";
   scale = "1.3 1.3 1.9";
   interiorFile = "~/data/interiors/pillar.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(pillarRight) {
   position = "77.6 -430.7 -1.2";
   rotation = "1 0 0 0";
   scale = "1.3 1.3 2";
   interiorFile = "~/data/interiors/pillar.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(pillarUpLeft) {
   position = "77.6 -424.8 6";
```

```
      rotation = "1 0 0 90";
      scale = "1.3 1.3 2.2";
      interiorFile = "~/data/interiors/pillar.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(pillarUpRight) {
      position = "77.6 -431.7 6";
      rotation = "1 0 0 90";
      scale = "1.3 1.3 2.2";
      interiorFile = "~/data/interiors/pillar.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
};
new WaterBlock(lake) {
   position = "0 -384 -19";
   rotation = "1 0 0 0";
   scale = "64 32 20";
   UseDepthMask = "0";
   surfaceTexture = "~/data/water/waterblack.jpg";
   ShoreTexture = "~/data/water/waterblack.jpg";
   envMapOverTexture = "~/data/water/waterblack.jpg";
   envMapUnderTexture = "~/data/water/waterblack.jpg";
   liquidType = "Water";
   density = "0.5";
   viscosity = "15";
   waveMagnitude = "0";
   surfaceOpacity = "0.5";
   envMapIntensity = "0.4";
   TessSurface = "50";
   TessShore = "60";
   SurfaceParallax = "0.5";
   FlowAngle = "0";
   FlowRate = "0";
   DistortGridScale = "0.1";
   DistortMag = "0.05";
   DistortTime = "0.5";
   ShoreDepth = "1";
   DepthGradient = "1";
   MinAlpha = "0.1";
   MaxAlpha = "1";
   tile = "1";
   removeWetEdges = "0";
   specularColor = "1 1 1 1";
   specularPower = "6";
      seedPoints = "0 0 1 0 1 1 0 1";
      textureSize = "32 32";
      floodFill = "1";
      params1 = "0.63 -2.41 0.33 0.21";
      envMapTexture = "~/data/skies/sunset_0007";
      params0 = "0.32 -0.67 0.066 0.5";
      params3 = "1.21 -0.61 0.13 -0.33";
      params2 = "0.39 0.39 0.2 0.133";
      Extent = "100 100 10";
};
new MissionArea(MissionArea) {
   Area = "-360 -648 720 1296";
   flightCeiling = "300";
   flightCeilingRange = "20";
      locked = "true";
```

```
};
new Sun() {
   azimuth = "0";
   elevation = "35";
   color = "0.988 0.985 0.98 1";
   ambient = "0.5 0.5 0.5 1";
      position = "0 0 0";
      direction = "0.57735 0.57735 -0.57735";
      locked = "true";
      rotation = "1 0 0 0";
      scale = "1 1 1";
};
new TerrainBlock(Terrain) {
   rotation = "1 0 0 0";
   scale = "1 1 1";
   detailTexture = "~/data/terrains/grass.jpg";
   terrainFile = "./MuseumMission.ter";
   squareSize = "8";
   emptySquares = "99744 443522 443778 444034";
   bumpScale = "1";
   bumpOffset = "0.01";
   zeroBumpScale = "8";
   tile = "1";
      position = "-1024 -1024 0";
      locked = "true";
};
new SimGroup(building) {

   new InteriorInstance(bars1) {
      position = "141.5 -418.8 15.5908";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/bars1.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(OutsideStairs) {
      position = "87.5261 -424.366 14";
      rotation = "1 0 0 0";
      scale = "1.4 1 1";
      interiorFile = "~/data/interiors/escalera.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(platform) {
      position = "32.1658 -362.8 0.05";
      rotation = "1 0 0 0";
      scale = "0.5 1.25 0.5";
      interiorFile = "~/data/interiors/platform.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(bars) {
      position = "131.45 -418.8 15.5908";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/bars.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(Museum) {
```

```
      position = "60 -400 12";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/museo1.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(house) {
      position = "45 -324.454 2";
      rotation = "0 0 1 180";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/chouse.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
};
new SimGroup(trees) {

   new TSStatic() {
      position = "-3.5022 -331.189 -0.6";
      rotation = "1 0 0 0";
      scale = "0.3 0.3 0.3";
      shapeName = "~/data/shapes/trees/tree.dts";
   };
   new TSStatic() {
      position = "-4.7409 -413.387 -0.6";
      rotation = "1 0 0 0";
      scale = "0.3 0.3 0.25";
      shapeName = "~/data/shapes/trees/tree3.dts";
   };
   new TSStatic() {
      position = "56.4162 -351.749 0.9789";
      rotation = "1 0 0 0";
      scale = "0.4 0.4 0.4";
      shapeName = "~/data/shapes/trees/tree3.dts";
   };
   new TSStatic() {
      position = "-3.34992 -354.717 0.5789";
      rotation = "1 0 0 0";
      scale = "0.3 0.3 0.2";
      shapeName = "~/data/shapes/trees/tree.dts";
   };
   new TSStatic() {
      position = "12.9579 -397.742 -0.89677";
      rotation = "1 0 0 0";
      scale = "0.3 0.3 0.3";
      shapeName = "~/data/shapes/trees/tree3.dts";
   };
   new TSStatic() {
      position = "43.1579 -451.142 -1.49598";
      rotation = "1 0 0 0";
      scale = "0.4 0.4 0.4";
      shapeName = "~/data/shapes/trees/tree2.dts";
   };
   new TSStatic() {
      position = "48.9853 -386.643 0.57891";
      rotation = "1 0 0 0";
      scale = "0.4 0.4 0.4";
      shapeName = "~/data/shapes/trees/tree.dts";
   };
   new TSStatic(tree2) {
```

```
      position = "22.4578 -441.532 -0.82109";
      rotation = "1 0 0 0";
      scale = "0.4 0.4 0.4";
      shapeName = "~/data/shapes/trees/tree2.dts";
   };
};
new SimGroup(InfoOfficeObjects) {

   new InteriorInstance(pillarSeat) {
      position = "33.5 -80 0.2";
      rotation = "0 0 -1 90";
      scale = "14 14 1";
      interiorFile = "~/data/interiors/pillarseat.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new StaticShape(officeBarLL) {
      position = "40.3328 -312.333 4.065";
      rotation = "0 0 -1 20.6266";
      scale = "0.1 2.55 0.05";
      dataBlock = "InfoDeskItem";
   };
   new StaticShape(officeBarRR) {
      position = "43.6791 -312.342 4.065";
      rotation = "0 0 1 19";
      scale = "0.1 2.55 0.05";
      dataBlock = "InfoDeskItem";
   };
   new StaticShape(officeBarLeft) {
      position = "41.0017 -313.239 4.065";
      rotation = "0 0 -1 52";
      scale = "0.1 1.8 0.05";
      dataBlock = "InfoDeskItem";
   };
   new StaticShape(officeBarRight) {
      position = "43.1339 -313.178 4.065";
      rotation = "0 0 1 235";
      scale = "0.1 1.8 0.05";
      dataBlock = "InfoDeskItem";
   };
   new StaticShape(officeBawr) {
      position = "42 -313.5 4.065";
      rotation = "1 0 0 0";
      scale = "2.3 0.1 0.05";
      dataBlock = "InfoDeskItem";
   };
   new InteriorInstance(pillarDesk) {
      position = "178 -311.5 1.95";
      rotation = "1 0 0 0";
      scale = "8 8 0.72";
      interiorFile = "~/data/interiors/pillardsk.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
};
new SimGroup(Paths) {

   new Path(Path22) {
      isLooping = "1";

      new Marker(m20) {
```

```
      position = "85.2751 -421.57 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "1";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
   new Marker(m21) {
      position = "85.8124 -413.541 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "2";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
};
new Path(Path23) {
   isLooping = "1";

   new Marker(m30) {
      position = "85.6155 -437.957 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "1";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
   new Marker(m31) {
      position = "85.6597 -431.6 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "2";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
};
new Path(Path24) {
   isLooping = "1";

   new Marker(m40) {
      position = "85.5346 -449.118 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "1";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
   new Marker(m41) {
      position = "85.8022 -442.424 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "2";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
```

```
};
new Path(Path25) {
   isLooping = "1";

   new Marker(m50) {
      position = "57.6149 -448.576 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "1";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
   new Marker(m51) {
      position = "62.7151 -447.392 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "2";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
};
new Path(Path26) {
   isLooping = "1";

   new Marker(m60) {
      position = "58.6901 -424.632 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "1";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
   new Marker(m61) {
      position = "58.9225 -431.519 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "2";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
};
new Path(Path27) {
   isLooping = "1";

   new Marker(m70) {
      position = "58.3187 -409.624 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "1";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
   new Marker(m71) {
      position = "58.5 -416.905 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
```

```
      seqNum = "2";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
};
new Path(Path21) {
   isLooping = "1";

   new Marker(m10) {
      position = "61.5518 -398.659 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "1";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
   new Marker(m11) {
      position = "66.23 -398.398 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "2";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
};
new Path(Path28) {
   isLooping = "1";

   new Marker(m80) {
      position = "58.7112 -397.882 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "1";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
   new Marker(m81) {
      position = "58.2955 -404.046 12.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "2";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
};
new Path(Path16) {
   isLooping = "1";

   new Marker(mar06) {
      position = "75.5287 -443.933 0.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "1";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
```

```
   };
   new Marker(mar16) {
      position = "65.3893 -435.311 0.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "2";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
};
new Path(Path15) {
   isLooping = "1";

   new Marker(mar05) {
      position = "83.3985 -442.801 0.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "1";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
   new Marker(mar15) {
      position = "72.1104 -435.008 0.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "2";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
};
new Path(Path14) {
   isLooping = "1";

   new Marker(mar04) {
      position = "91.041 -442.34 0.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "1";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
   new Marker(mar24) {
      position = "81.4228 -434.671 0.5";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      seqNum = "2";
      type = "Normal";
      msToNext = "1000";
      smoothingType = "Spline";
   };
};
new Path(Path12) {
   isLooping = "1";

   new Marker(mar02) {
      position = "84.3906 -403.871 0.5";
      rotation = "1 0 0 0";
```

```
         scale = "1 1 1";
         seqNum = "1";
         type = "Normal";
         msToNext = "1000";
         smoothingType = "Spline";
      };
      new Marker(mar12) {
         position = "73.0148 -412.685 0.5";
         rotation = "1 0 0 0";
         scale = "1 1 1";
         seqNum = "2";
         type = "Normal";
         msToNext = "1000";
         smoothingType = "Spline";
      };
   };
   new Path(Path13) {
      isLooping = "1";

      new Marker(mar03) {
         position = "91.7309 -406.121 0.5";
         rotation = "1 0 0 0";
         scale = "1 1 1";
         seqNum = "1";
         type = "Normal";
         msToNext = "1000";
         smoothingType = "Spline";
      };
      new Marker(mar13) {
         position = "79.4785 -413.116 0.5";
         rotation = "1 0 0 0";
         scale = "1 1 1";
         seqNum = "2";
         type = "Normal";
         msToNext = "1000";
         smoothingType = "Spline";
      };
   };
   new Path(Path11) {
      isLooping = "0";

      new Marker(mar01) {
         position = "79.5781 -403.642 0.5";
         rotation = "1 0 0 0";
         scale = "1 1 1";
         seqNum = "1";
         type = "Normal";
         msToNext = "1000";
         smoothingType = "Spline";
      };
      new Marker(mar11) {
         position = "65.793 -412.281 0.5";
         rotation = "1 0 0 0";
         scale = "1 1 1";
         seqNum = "2";
         type = "Normal";
         msToNext = "1000";
         smoothingType = "Spline";
      };
   };
};
```

```
new SimGroup(scales) {

   new InteriorInstance(scaleR2) {
      position = "98.673 -442.5 6";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/scalesss.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(scaleL1) {
      position = "94 -392.05 6";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/scalesss.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(scaleL2) {
      position = "98.1 -392.05 6";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/scalesss.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(scaleL3) {
      position = "102.2 -392.05 6";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/scalesss.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(scaleL4) {
      position = "106.4 -392.05 6";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/scalesss.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(scaleL5) {
      position = "110.6 -392.05 6";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/scalesss.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(scaleR1) {
      position = "94 -442.5 6";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/scalesss.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance(scaleR3) {
      position = "102.2 -442.5 6";
      rotation = "1 0 0 0";
```

```
   scale = "1 1 1";
   interiorFile = "~/data/interiors/scalesss.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(scaleR4) {
   position = "106.3 -442.5 6";
   rotation = "1 0 0 0";
   scale = "1 1 1";
   interiorFile = "~/data/interiors/scalesss.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(scaleR5) {
   position = "110.4 -442.5 6";
   rotation = "1 0 0 0";
   scale = "1 1 1";
   interiorFile = "~/data/interiors/scalesss.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(scaleR6) {
   position = "114.5 -442.5 6";
   rotation = "1 0 0 0";
   scale = "1 1 1";
   interiorFile = "~/data/interiors/scalesss.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(scaleR7) {
   position = "118.6 -442.5 6";
   rotation = "1 0 0 0";
   scale = "1 1 1";
   interiorFile = "~/data/interiors/scalesss.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(scaleL6) {
   position = "114.8 -392.05 6";
   rotation = "1 0 0 0";
   scale = "1 1 1";
   interiorFile = "~/data/interiors/scalesss.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(scaleL7) {
   position = "119 -392.05 6";
   rotation = "1 0 0 0";
   scale = "1 1 1";
   interiorFile = "~/data/interiors/scalesss.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
new InteriorInstance(scaleL8) {
   position = "130 -392.05 6";
   rotation = "1 0 0 0";
   scale = "1.2 1 1";
   interiorFile = "~/data/interiors/scalesss.dif";
   useGLLighting = "0";
   showTerrainInside = "0";
};
```

```
      new InteriorInstance(scaleR8) {
         position = "133 -442.5 6";
         rotation = "1 0 0 0";
         scale = "1.3 1 1";
         interiorFile = "~/data/interiors/scalesss.dif";
         useGLLighting = "0";
         showTerrainInside = "0";
      };
   };
   new InteriorInstance() {
      position = "91.9 -425.5 14.7";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/bb.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance() {
      position = "91.9 -421.9 14.7";
      rotation = "1 0 0 0";
      scale = "1 1 1";
      interiorFile = "~/data/interiors/bb.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
   new InteriorInstance() {
      position = "45.5 -312.2 3.88";
      rotation = "1 0 0 0";
      scale = "2 0.05 0.01";
      interiorFile = "~/data/interiors/cube.dif";
      useGLLighting = "0";
      showTerrainInside = "0";
   };
};
//--------------------------------------------------------------------
```

ΒΙΒΛΙΟΓΡΑΦΙΑ ΚΑΙ ΙΣΤΟΣΕΛΙΔΕΣ

-Kenneth C. Finney, 3D Game Programming all in one, Premier Press 2004
-Bending Stang, Game Engines, features and possibilities, IMM DTU 2003
-http://www.garagegames.com
-http://www.planetquake.com
-http://www.hallofworlds.com
-http://www.psionic.com
-http://www.bravetree.com
-http://www.codesampler.com
-http://www.etse.urv.es
-http://dynamic.gamespy.com
-http://www.obsidiangames.com