ΑΝΩΤΑΤΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ
(Α.Τ.Ε.Ι.)ΚΡΗΤΗΣ
ΠΑΡΑΡΤΗΜΑ ΡΕΘΥΜΝΟΥ
ΤΜΗΜΑ ΜΟΥΣΙΚΗΣ ΤΕΧΝΟΛΟΓΙΑΣ & ΑΚΟΥΣΤΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΛΕΙΤΟΥΡΓΙΑ ΤΩΝ AUDIO API : ΤΟ AUDIO API OPENAL

ΓΡΗΓΟΡΑΤΟΥ ΣΟΦΙΑ Α.Μ.366

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ - ΔΙΑΜΑΝΤΟΠΟΥΛΟΣ ΤΑΞΙΑΡΧΗΣ

ΡΕΘΥΜΝΟ 2008

ΠΕΡΙΕΧΟΜΕΝΑ

Σελίδες

ΕΙΣΑΓΩΓΗ

1. ΑΝΤΙΚΕΙΜΕΝΟ ΤΗΣ ΠΤΥΧΙΑΚΗΣ

Αντικείμενο της παρούσας πτυχιακής, είναι η εισαγωγή στην λειτουργία των Audio API. Ένα API (Application Programming Interface) θα μπορούσε να οριστεί ως ένα σύνολο από υποπρογράμματα - ρουτίνες κλήσης, βιβλιοθήκες ή προγραμματιστικά αντικείμενα, τα οποία παρέχουν στον προγραμματιστή προγραμματιστικά βοηθήματα.

Στην πτυχιακή εργασία γίνεται παρουσίαση των Audio API γενικότερα, ενώ επικεντρώνεται στη δομή και λειτουργία του OpenAL. Το API OpenAL είναι ένα Open Source (ανοικτού κώδικα), cross-platform (ανεξάρτητης πλατφόρμας), 3D audio API κατάλληλο για αναπαραγωγή ήχου και μουσική, καθώς επίσης και για μια σειρά εφαρμογών για 3D Audio, όπως η επένδυση ήχου σε παιχνίδια για υπολογιστή ή για εξειδικευμένες πλατφόρμες παιχνιδιών.

Το OpenAL επιλέχθηκε μετά από εξέταση των Audio API για την γλώσσα C ως το επικρατέστερο μιας και αποτελεί δημιουργία της εταιρίας Creative Labs και υποστηρίζεται πλήρως από την Creative και την NVIDIA audio devices, οι οποίες καταλαμβάνουν το μεγαλύτερο μερίδιο της αγοράς.

Το OpenAL Audio API έχει μεγάλες ομοιότητες στον τρόπο προγραμματισμού με το API OpenGL, το οποίο έχει καθιερωθεί εδώ και καιρό από τις εταιρίες παιχνιδιών για την δημιουργία γραφικών. Η ταυτόχρονη χρήση OpenAL και OpenGL αποτελούν ένα πλήρες πακέτο για την δημιουργία γραφικών και ήχου σε παιχνίδια.

2. ΔΙΑΡΘΡΩΣΗ ΤΗΣ ΠΤΥΧΙΑΚΗΣ

Η υπόλοιπη πτυχιακή εργασία ξεκινάει με μια παρουσίαση του απαραίτητου θεωρητικού υπόβαθρου, εξηγώντας, αρχικά (Κεφαλαίο 1), τις βασικές έννοιες των API και Audio API γενικότερα, καθώς και τις βασικές λειτουργίες τους . Στο τέλος του κεφαλαίου αυτού παρουσιάζονται τα γνωστότερα Audio API.

Το Κεφαλαίο 2 επικεντρώνεται στη δομή και λειτουργία του OpenAL. Αρχικά εισάγονται οι βασικές έννοιες, ώστε να γίνει εφικτή η χρήση του και καταλήγει σε πιο εξειδικευμένα υποκεφάλαια προγραμματισμού με την βοήθεια του συγκεκριμένου API. Στο Κεφαλαίο 3 παρατίθενται ολοκληρωμένες υλοποιήσεις υπό μορφή κώδικα, με την χρήση της OpenAL, που πραγματοποιήθηκαν κατά την διάρκεια της εκπόνησης της πτυχιακής εργασίας.

Στο Παράρτημα Α δίνεται ένας πλήρης κατάλογος εντολών, ώστε να είναι δυνατή η περαιτέρω εμβάθυνση στο συγκεκριμένο αντικείμενο, ενώ στο Παράρτημα Β δίδονται απαραίτητες πληροφορίες εγκατάστασης (Compiler/IDE και API OpenAL).

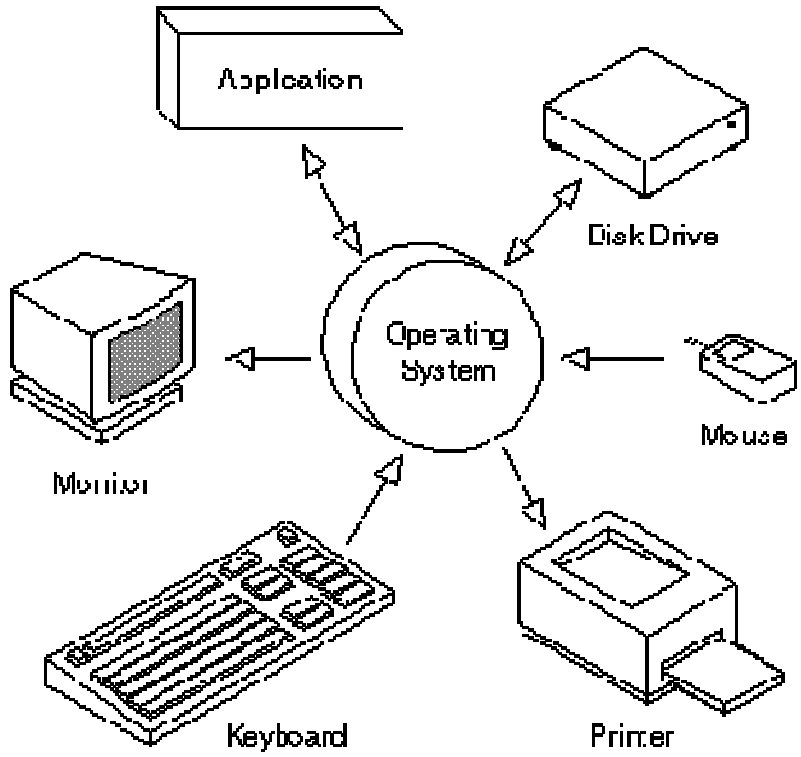
Το CD, το οποίο συνοδεύει την παρούσα πτυχιακή εργασία,περιλαμβάνει:

- Τον υλοποιήσιμο κώδικα υπό μορφή source.
- Τα αρχεία εκτέλεσης του κώδικα (.exe, .dll κλπ).

ΚΕΦΑΛΑΙΟ 1
ΘΕΩΡΗΤΙΚΗ ΕΙΣΑΓΩΓΗ – AUDIO API

1. ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΛΕΙΤΟΥΡΓΙΑ ΤΩΝ API

Η επικοινωνία μιας εφαρμογής με μια οποιαδήποτε συσκευή γίνεται μέσω ενός ειδικού λογισμικού, το οποίο συνήθως εμπεριέχεται στο λειτουργικό σύστημα. Το λογισμικό αυτό αποτελεί ένα software interface, δηλαδή μια διεπαφή λογισμικού ανάμεσα στις εφαρμογές που γράφονται από προγραμματιστές για τις συσκευές. Μπορούμε να μιλήσουμε για δύο επίπεδα επικοινωνίας: το ένα είναι τα γνωστά σε όλους μας *προγράμματα καθοδήγησης (drivers)* και το άλλο είναι τα *API (Application Programming Interface)*. [1]



Εικόνα 1 Το Λειτουργικό Σύστημα - Software Interface.

Ένα drive, ή πρόγραμμα καθοδήγησης ή οδηγός συσκευών, είναι ένα πρόγραμμα που επιτρέπει σε υψηλότερου επιπέδου προγράμματα να αλληλεπιδράσουν με μια συσκευή και το υλικό του υπολογιστή. Απλοποιεί τον προγραμματισμό δρώντας ως μεταφραστής μεταξύ μιας συσκευής και των εφαρμογών. Έτσι υψηλότερου επιπέδου εφαρμογές μπορεί να δημιουργηθούν, ανεξάρτητα από το υλικό του υπολογιστή που θα λειτουργήσουν. Οι περισσότερες εφαρμογές έχουν πρόσβαση στις συσκευές με την χρήση υψηλού επιπέδου γενικών εντολών. Ο *driver* δέχεται αυτές τις γενικές δηλώσεις και τις μετατρέπει στις χαμηλού επιπέδου εντολές που απαιτούνται από τη συσκευή. [2]

Ένα API θα μπορούσε να οριστεί ως ένα σύνολο πρωτοκόλλων, υποπρογραμμάτων (ρουτίνες κλήσης), βιβλιοθηκών ή προγραμματιστικών αντικείμενων και εργαλείων για τις εφαρμογές λογισμικού, το οποίο παρέχει στον προγραμματιστή προγραμματιστικά βοηθήματα. Τέτοιου είδους βοηθήματα, αποτελούν έτοιμες ρουτίνες, οι οποίες παρέχουν διάφορες δυνατότητες, ανάλογα με την γλώσσα προγραμματισμού που χρησιμοποιείται. Η λέξη API προέρχεται από τα ακρώνυμα των λέξεων Application Programming Interface (Προγραμματιστικές Διεπαφές Εφαρμογών). [1,3]

Ο τρόπος με τον οποίο επικοινωνεί μια εφαρμογή (δηλαδή στην ουσία οι χρήστες της εφαρμογής) με τους driver, γίνεται μέσω ενός API. Ένα API μπορεί επίσης να χρησιμοποιηθεί από μια εφαρμογή για να επικοινωνήσει με το λειτουργικό σύστημα, με άλλα προγράμματα ελέγχου, όπως συστήματα διαχείρισης βάσεων δεδομένων (DBMS), με τη μνήμη, με αρχεία, με πρωτόκολλα επικοινωνιών και με συστήματα δικτύωσης. [1,3,4,5]

Σε αντίθετη περίπτωση ο προγραμματιστής θα έπρεπε να γράψει κώδικα από την αρχή για να μπορεί να ελέγξει κάποιες κοινές λειτουργίες. Κάθε API παρέχει τις δικές του βιβλιοθήκες, μέσα από τις οποίες ο προγραμματιστής μπορεί να επιλέξει όποιες επιθυμεί. Έτσι, όταν προγραμματίζεται μία εφαρμογή, επιλέγεται ένα συγκεκριμένο API, ώστε να υπάρχει αποτελεσματικότερη πρόσβαση και καλύτερος έλεγχος στους πόρους του συστήματος (software ή hardware resources). Ο προγραμματισμός με την βοήθεια των API υπονοεί ότι κάποια ενότητα προγράμματος, είτε είναι αυτόνομα διαθέσιμη στον υπολογιστή προκειμένου να εκτελεστεί κάποια λειτουργία, είτε εμπεριέχεται σε κάποια βιβλιοθήκη η οποία πρέπει να συμπεριληφθεί μαζί με τον υπόλοιπο κώδικα. Και στις δύο περιπτώσεις εκτελούνται κλήσεις, οι οποίες ανασύρουν τον αντίστοιχο κώδικα του API. [1,6]

Επίσης πολλοί τύποι λειτουργικών συστημάτων εμπεριέχουν APIs για τα συστήματα γραφικής παράστασης, για βάσεις δεδομένων, για δικτυακές εφαρμογές, για τις υπηρεσίες Ιστού ή ακόμη και για παιχνίδια υπολογιστών. Λειτουργικά συστήματα, όπως τα Microsoft Windows ή το Linux, για παράδειγμα, παρέχουν τέτοιες βιβλιοθήκες, έτσι ώστε οι προγραμματιστές να μπορούν να γράψουν εφαρμογές σύμφωνες με το λειτουργικό περιβάλλον [1,3,4,5]

Εικόνα 2

Αν και τα APIs σχεδιάζονται κυρίως για τους προγραμματιστές, αποτελούν τελικά ένα καλό εργαλείο και για τους χρήστες, επειδή εγγυώνται ότι όλα τα προγράμματα που χρησιμοποιούν ένα κοινό API, θα έχουν παρόμοιες διεπαφές σε επίπεδο περιβάλλοντος χρήσης. Αυτό διευκολύνει τους χρήστες να χειρίζονται με μεγαλύτερη ευκολία νέα προγράμματα. [1,3]

Ένα API που δεν απαιτεί δικαιώματα για την πρόσβαση και τη χρήση καλείται "Open ή Open source." Τα APIs που παρέχονται από Free software (όπως όλο το λογισμικό που διανέμεται με άδεια GNU[1]) είναι ανοικτά εξ ορισμού, δεδομένου ότι ο καθένας μπορεί να εξετάσει τον κώδικα πηγής του λογισμικού, αν και συνήθως οι προδιαγραφές χρήσεις ποικίλουν για κάθε API. 4,7

Εικόνα 3 Gnu - General Public License : Το σήμα της άδειας Gnu

Δύο γενικές γραμμές πολιτικών υπάρχουν σχετικά με την έκδοση APIs:

- Μερικές επιχειρήσεις περιφρουρούν με ζήλο τα APIs τους. Παραδείγματος χάριν, η Sony για να δημιουργήσει το επίσημο PlayStation2 χρησιμοποίησε API, τα οποία είναι διαθέσιμα μόνο στους εξουσιοδοτημένους υπεύθυνους για την ανάπτυξη PlayStation. Αυτό συνέβη επειδή η Sony θέλησε να έχει τον έλεγχο όσων γράφουν παιχνίδια για την πλατφόρμα του PlayStation 2, προκειμένου να ωφεληθεί από αυτούς όσο το δυνατόν περισσότερο. Αποτέλεσμα αυτής της πολιτικής ήταν να ζημιωθεί η συγκεκριμένη εταιρία, αφού οι προγραμματιστές παιχνιδιών προτίμησαν κονσόλες με API ανοικτού κώδικα. Το PlayStation 3 είναι βασισμένο εξ ολοκλήρου σε ανοικτό και δημόσια διαθέσιμο APIs.4

- Άλλες επιχειρήσεις διαδίδουν ελεύθερα τα APIs τους, αφού οι προγραμματιστές παιχνιδιών προτίμησαν κονσόλες με API ανοικτού κώδικα και έτσι όλο και περισσότερος κόσμος αγοράζει τις αντίστοιχες κονσόλες.4

[1] Gnu - General Public License : http://www.gnu.org/ (Πρόσβαση : 4 Οκτωβρίου 2006)

2. SOFTWARE DEVELOPMENT KIT –SDK

Ένα *software development kit (SDK ή "devkit")* είναι ένα σύνολο εργαλείων ανάπτυξης, που παρέχουν σε έναν προγραμματιστή την δυνατότητα ανάπτυξης εφαρμογών για ένα συγκεκριμένο πακέτο λογισμικού (*software package*) ή μια πλατφόρμα υλικού (*hardware platform*) ή γενικότερα για ένα λειτουργικό σύστημα. [4,8,9,10]

Μπορεί να είναι κάτι τόσο απλό όσο ένα API ή να περιλαμβάνει περίπλοκο υλικό για να επικοινωνήσει με ένα συγκεκριμένο *ενσωματωμένο σύστημα (embedded system*[1]*),* καθιστώντας έτσι τους όρους SDK και API τελείως ανεξάρτητους. Γενικά ένα *SDK* περιλαμβάνει εργαλεία *εκσφαλμάτωσης-διόρθωσης σφαλμάτων (Debugging),* όπως ένα *IDE*[2] καθώς και άλλες λειτουργίες. Επίσης συχνά περιλαμβάνει δείγματα κώδικα και βοηθητικές τεχνικές σημειώσεις για να διευκρινίσει τα σημεία των λειτουργιών του υλικού. [4,8]

Πολλά SDKs παρέχονται δωρεάν για να ενθαρρύνουν τους υπεύθυνους για την ανάπτυξη λογισμικού (*software engineer*) να χρησιμοποιήσουν το σύστημα ή την γλώσσα. Μερικές φορές αυτό χρησιμοποιείται ως εργαλείο μάρκετινγκ. Παραδείγματος χάριν μια εταιρία παρέχει το SDK της δωρεάν, ώστε να χρησιμοποιηθεί και να δοκιμαστεί από διάφορους ανθρώπους. Στη συνέχεια περισσότερος κόσμος θα ενθαρρυνθεί για να αγοράσει το προϊόν που απευθύνεται στο συγκεκριμένο SDK της, δεδομένου ότι μπορούν να τα προγραμματίσουν δωρεάν. [8]

Πρέπει να σημειωθεί ότι οι προμηθευτές SDKs για συγκεκριμένα συστήματα ή υποσυστήματα μπορούν μερικές φορές να αντικαταστήσουν τον όρο software με έναν πιο συγκεκριμένο. Παραδείγματος χάριν, και η Microsoft και η Apple παρέχουν *Driver Development Kits (DDK)* για την ανάπτυξη των οδηγών συσκευών (*device drivers*), ενώ τα εμπορικά σήματα *Palm Source* παρέχουν *PalmOS Development Kit (PDK)*. [8]

Σε πολλές περιπτώσεις, ένα API αποτελεί μέρος ενός *Software development kit - SDK ή "devkit"),* αλλά ακόμα και στην περίπτωση που δεν συνοδεύεται από ένα *SDK* παρέχει κάποια βασικά αρχεία, τα οποία είναι απαραίτητα, ώστε να μπορούν οι προγραμματιστές να εργαστούν με αυτό. Σε πολλές περιπτώσεις πολύ γνωστά API, που μερικές φορές αποτελούν και standard, βρίσκονται ενσωματωμένα με τους μεταγλωττιστές και παρέχονται μαζί με αυτούς. Παραδείγματα τέτοια είναι η OpenGL, ή το SDL [4,8,11]

[1] Embedded System: Αντίθετα από έναν γενικής χρήσης υπολογιστή, όπως ένας προσωπικός υπολογιστής, ένα ενσωματωμένο σύστημα (embedded system) εκτελεί προκαθορισμένες λειτουργίες, οι οποίες συνήθως αφορούν πολύ συγκεκριμένες απαιτήσεις χρήσης. Ένα φορητό GPS σύστημα ή ένα PDA (Personal Digital Assistant) θεωρούνται γενικά ενσωματωμένες συσκευές λόγω της φύσης της αρχιτεκτονικής τους , ακόμα κι αν είναι πιο επεκτάσιμες σε όρους λογισμικού. [13]

[2] Integrated development Environment (IDE), γνωστό και ως Integrated Design Environment ή Integrated Debugging Environment - Ενσωματωμένο περιβάλλον διόρθωσης: είναι ένας τύπος λογισμικού υπολογιστών, που βοηθά τους προγραμματιστές υπολογιστών στην ανάπτυξη λογισμικού. Ένα IDE αποτελείται συνήθως από έναν source code editor, έναν μεταγλωττιστή ή/και έναν διερμηνέα, τα εργαλεία κατασκευής-αυτοματοποίησης, και (συνήθως) έναν διορθωτή. [13]

Η βασική δομή ενός API για C ακολουθεί τη γενικότερη δομή της C, περιέχει δηλαδή τα εξής αρχεία :

- *Include files ή Header files* : Συνήθως περιέχονται σε έναν φάκελο με το όνομα include. Αυτά τα αρχεία έχουν την γνωστή κατάληξη .h και είναι τα αρχεία που μπορεί να καλέσει ένας προγραμματιστής μέσα στον κώδικα του, ώστε να χρησιμοποιήσει τις αντίστοιχες εντολές που παρέχονται από αυτά. [14]

- *Library files:* Συνήθως περιέχονται στον φάκελο με όνομα lib και γενικά έχουν κατάληξη .lib. Αυτά τα αρχεία περιέχουν τμήματα κώδικα που χρησιμοποιούν τα αρχεία Include όταν καλούνται. [15]

- *Dynamic Link Library files*: Συνήθως περιέχονται στον φάκελο με όνομα bin και έχουν κατάληξη .dll. Και αυτά τα αρχεία περιέχουν τμήματα κώδικα, που θα χρησιμοποιηθούν από τα Include files. Τα αρχεία αυτά, σε ορισμένες περιπτώσεις, μπορεί να διαφέρουν για Debug και Release και πρέπει να τοποθετούνται είτε στο φάκελο του εκτελέσιμου .exe που δημιούργησε ο κώδικας, είτε μέσα στο windows system file. Αυτά τα αρχεία είναι απαραίτητα και κατά τον προγραμματισμό μιας εφαρμογής, αλλά και κατά την εκτέλεση της από έναν χρήστη. [16]

Αυτοί οι τρεις τύποι αρχείων συνεργάζονται μεταξύ τους κάθε φορά που εκτελείται ένα τμήμα κώδικα.

Αν ανατρέξουμε στο παράδειγμα που ακολουθεί, το πρώτο πράγμα που θα δούμε είναι η γραμμή #include <stdio.h>. Μέσα σε αυτό το Include files υπάρχει η εντολή printf και ο τρόπος σύνταξης της. Αυτή είναι ουσιαστικά μια συνάρτηση, που μπορούμε να χρησιμοποιήσουμε όσες φορές επιθυμούμε. Ο κώδικας της συνάρτησης αυτής υπάρχει μέσα σε κάποιο αρχείο .llb ή dll που ο μεταγλωττιστής γνωρίζει και ανασύρει κάθε φορά που χρησιμοποιείται.

```
#include <stdio.h>
int main(void) {
printf("Hello, world C!\n");
return 0;
}
```

3. ΤΑ ΓΝΩΣΤΟΤΕΡΑ AUDIO API ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΟΥΝΤΑΙ ΣΗΜΕΡΑ

Οι ικανότητες των sound cards και sound APIs έχουν αυξηθεί κατά τη διάρκεια των ετών. Ο τρισδιάστατος ήχος αποτελεί πλέον ένα σημαντικό στοιχείο στα παιχνίδια και στις υπόλοιπες εφαρμογές, που μπορεί να τα θέσει σε ένα νέο επίπεδο. [17]

Διαφορετικά programming APIs έχουν αναπτυχθεί για αυτόν τον σκοπό, όπως το *DirectSound3D, PortAudio,Boss, libsndfile,OpenAL*. [17]

3.1 DirectSound3D

Το *DirectSound3D* είναι ένα τμήμα λογισμικού της βιβλιοθήκης *DirectX,* που παρέχεται από τη *Microsoft*.Δεν είναι ένα *cross-platform API* μιας και απευθύνεται σε συστήματα που λειτουργούν με το λογισμικό της *Microsoft,* σε γλώσσα *C\C++*.[18,19]

Το *DirectSound3D (DS3D)* είναι μια προσθήκη στο σύστημα *DirectX* της *Microsoft,* που προορίζεται να τυποποιήσει τον τρισδιάστατο ήχο της *Microsoft Windows*. Το *DS3D* εισήχθη με το *DirectX 3* το 1996. Παρέχει απ' ευθείας επικοινωνία μεταξύ των εφαρμογών και των καρτών ήχου, επιτρέποντας στις εφαρμογές να παράγουν ήχους και μουσική. [18]

Εικόνα 4 Microsoft DirectX 9

Μετά από πολλά έτη ανάπτυξης, σήμερα το *DirectSound* είναι ένα πολύ ώριμο API και παρέχει πολλά χρήσιμα εργαλεία, όπως η δυνατότητα να παιχτεί πολυκάναλος ήχος, μίξη του ήχου, καθώς και διάφορα effect, όπως reverb, echo, flange κ.λ.π., ενώ επίσης κάνει χρήση των hardware buffers. [18]

Ενώ το *DirectSound* είχε ως αρχικό σκοπό να χρησιμοποιηθεί από τα παιχνίδια, διάφορες επαγγελματικές ακουστικές εφαρμογές εκμεταλλεύονται πλέον τις πολλές και διαφορετικές δυνατότητές του. [18]

3.2 PortAudio

Η *PortAudio* είναι μια ελεύθερη *cross platform, open-source, audio I/O library*. Είναι κατασκευασμένη για προγράμματα στην γλώσσα C και έχει την δυνατότητα να τρέξει σε πολλές πλατφόρμες όπως: *Windows, MacOS (8,9,X), Unix (OSS), SGI, και BeOS.* Το PortAudio χρησιμοποιείται σαν βάση για τα περιβάλλοντα ήχου Audiomulch και Jsyn. [20,21, 22,23]

Η PortAudio παρέχει ένα πολύ απλό API, για καταγραφή και αναπαραγωγή ήχου. Χρησιμοποιεί μια απλή ρουτίνα *callback function(*λειτουργία επανάκλησης*)*, ενώ μπορεί να λειτουργήσει και ως *Real-Time Audio Library χρησιμοποιώντας stream αρχείων.* [20,21,24,25]

Η βασική της δομή περιλαμβάνει :

- *Audio devices* που αντιπροσωπεύουν τις εισόδους και εξόδους του ηχητικού συστήματος. Μέσω των audio devices παρέχεται δυνατότητα απαρίθμησης των διαθέσιμων συσκευών (καρτών ήχου), ενώ μπορεί να διαχειρισθεί ή να ανιχνεύσει ορισμένα χαρακτηριστηκά των συσκευών. [25]

- *Audio streams* που διαχειρίζονται τις ενεργές εισόδους και εξόδους των συσκευών, τα οποία ταυτόχρονα δύνανται να διαχειριστούν μόνο ένα device εισαγωγής και ένα device αναπαραγωγής. Τα streams μπορεί να είναι πλήρως αμφίδρομα ή ημιαμφίδρομα (full duplex or half duplex). [25]

3.3 Libsndfine

Η *Libsndfine* είναι μια *audio library* για C. Είναι *Open source API* με άδεια -*Gnu Lesser General Public License* και κατασκευάστηκε για περιβάλλον Linux, αλλά μπορεί να χρησιμοποιηθεί και να τρέξει σε οποιοδήποτε περιβάλλον Unix, καθώς και στα συστήματα Win32 και MacOS. Έχει συνταχθεί και εξεταστεί (κάποια στιγμή) στα ακόλουθα συστήματα: *586-pc-Linux-gnu (Linux on PC hardware), PowerPC-unknown-Linux-gnu (Linux on Apple Mac hardware), PowerPC-apple-Darwin7.0 (Mac OS X 10.3), Sparc-sun-Solaris2.8 (με χρήση GCC), Mips-sgi-irix5.3 (με χρήση GCC), QNX 6.0, I386-unknown-openbsd2.9 και Win32 (Microsoft Visual C++).* Προς το παρόν, κάθε νέα έκδοση λειτουργεί σε *Linux, PowerPC Linux, Mac OSX* σε PowerPC και Win32. [26]

Η *Libsndfine* έχει δυνατότητα ανάγνωσης και εγγραφής, σε ένα μεγάλο αριθμό από *file formats,* όπως *WAV, AIFF/AIFF-C ,Ogg Vorbis*, κ.λ.π, ενώ ο σχεδιασμός της *Libsndfine* κάνει εύκολη την δημιουργία νέων *sound file formats*. [26]

3.4 Bass

Η *Bass* είναι μια *audio library* για τα λειτουργικά συστήματα *Windows* και *Mac OSX*. Για τα *Windows* απαιτείται η χρήση *DirectX 3* ή ανώτερη έκδοση για την παραγωγή ήχου, καθώς και για την αξιοποίηση του υλικού από το *DirectSound* και *DirectSound3D*, όπου είναι διαθέσιμο. Στα *OSX* η *Bass* χρησιμοποιεί το *CoreAudio* για την παραγωγή ήχου και συνίσταται *OSX 10.3* ή ανώτερη έκδοση. Επίσης υποστηρίζονται από τα *PowerPC* και *Intel Macs*, ενώ παρέχονται APIs για *C/C++, Delphi, Visual Basic* και *MASM*. [27]

Πρέπει να σημειωθεί ότι η *Bass* διανέμεται ελεύθερα για ανάπτυξη σε μη εμπορεύσιμο λογισμικό, όμως για χρήση της σε εμπορεύσιμο λογισμικό απαιτείται η αγορά άδειας χρήσης. [27]

Εικόνα 5 Το λογότυπο της Bass

Ο σκοπός της είναι να παρέχει στους προγραμματιστές τον πιο δυνατό, αλλά και εύχρηστο τρόπο για δημιουργία *samples, stream αρχείων (MP3, MP2, MP1, OGG, WAV), MOD*[1] *(XM, IT, S3M, MOD, MTM, UMX), MOD3*[2] *(MP3/OGG),* καθώς και functions για αναπαραγωγή ήχου σε CD. Υποστηρίζει 3D προσδιορισμό θέσης για *samples*, *streams* και *MOD* με την υποστήριξη *του Aureal's A3D(Aureal 3-Dimensional)* και *Creative Lab's EAX (Environmental Audio eXtensions).* [27,28,29]

Εικόνα 6 A3D & EAX

[1] MOD: Είναι ένα file format που χρησιμοποιήθηκε αρχικά για να αντιπροσωπεύσει τον ίδιο τον ήχο μετατρέποντάς τον σε .mod αρχεία. Ήταν το πρώτο module file format, αν εξαιρεθεί η Amiga, που χρησιμοποιούσε αρχεία mod.echoing. Ένα mod αρχείο περιέχει ένα σύνολο *οργάνων(instruments)* υπό μορφή δειγμάτων και έναν ενδεικτικό αριθμό μοτίβων που δείχνουν πότε και πως τα δείγματα θα εκτελεστούν. Οι πιο πρόσφατες εκδόσεις του μπορούν να χρησιμοποιήσουν έως και 32 κανάλια και 31 όργανα, ενώ τα δείγματα αποθηκεύονται σε 8-bit PCM format. [30,31]

[2] MOD3: Είναι ένα file format, που περιλαμβάνει τα χαρακτηριστικά γνωρίσματα του MOD, αλλά με μια μεγάλη διαφορά στην συμπίεση δείγματος. Το MOD3 επιτρέπει την μέγιστη ποιότητα και συμπίεση σε MP3/OGG, επειδή η κωδικοποίηση MP3/OGG λειτουργεί καλύτερα με τα 16-bit δείγματα. [27]

3.5 OpenAL

Η *OpenAL* είναι *ένα Open Source cross-platform 3D audio API,* κατάλληλη για χρήση σε εφαρμογές παιχνιδιών και πολλών άλλων τύπων εφαρμογών ήχου. Υπεύθυνοι για την ανάπτυξη του είναι η Loki Entertainment και η Creative Labs. [32,33]

Υποστηρίζει τη χρήση *DirectSound3D*, Aureal's A3D*(Aureal 3-Dimensional)* και *EAX(Environmental Audio eXtensions).* Αξιοσημείωτο είναι ότι υπάρχουν λειτουργίες που περιλαμβάνονται στη OpenAL και επιτρέπουν τη χρήση αυτών των APIs χωρίς καθόλου επιπλέον προγραμματισμό.[34]

Η OpenAL είναι ακόμα σε στάδιο ανάπτυξης, από την Creative Labs. Αν και υπάρχει συνεχής ανάπτυξη API, ακόμα δεν αποτελεί μια πλήρη βιβλιοθήκη. Οι σχεδιαστές της OpenAL μιμήθηκαν το *API OpenGL,* που είναι ένα από τα καλύτερα API που σχεδιάσθηκαν και χρησιμοποιήθηκαν. Χρησιμοποιείται ήδη από παρά πολλά παιχνίδια και έχει καταπληκτικά αποτελέσματα. [32,33]

Επιτρέπει στους προγραμματιστές να φορτώσουν οποιονδήποτε ήχο επιθυμούν και να ελέγχουν ορισμένα βασικά χαρακτηριστικά του, όπως τη θέση, την ταχύτητα, την κατεύθυνση, καθώς και τις γωνίες που καθορίζουν πώς ο ήχος ταξιδεύει. Όλοι οι ήχοι τοποθετούνται σχετικά με τον ακροατή που αντιπροσωπεύει την τρέχουσα θέση. Οι ακουστικές έννοιες, όπως το panning και το left/right κανάλι, δεν υποστηρίζονται άμεσα, ενώ δεν υπάρχει καμία υποστήριξη για MIDI.[34, 35,36]

Εικόνα 7 Το λογότυπο της OpenAL.

Η OpenAL περιλαμβάνει συμβατές επεκτάσεις με το πρότυπο *IA-SIG (Interactive Audio Special Interest Group: http://www.iasig.org/*), το οποίο αφορά την δημιουργία 3D μοντέλων και, μεταξύ άλλων, περιλαμβάνει οδηγίες διαχείρισης για τον χειρισμό της κατευθυντικότητας των πηγών, την μείωση έντασης σε σχέση με την απόσταση (distance-related attenuation), καθώς επίσης και διάφορα effects χώρου όπως: ανάκλαση (reflection),ηχητική παρεμπόδιση (obstruction) , ηχητική διάδοση (transmission) και αντήχηση (reverberation). [35,36,37]

Εικόνα 8 Το λογότυπο της *IA-SIG*

3.6 Audio API και Vista

Τα πολύ αναμενόμενα Windows Vista έθεσαν νέα standard στο κομμάτι της επεξεργασίας ήχου. Σύμφωνα με την Microsoft οι αλλαγές αυτές έγιναν με βασικό σκοπό το υποσύστημα ήχου να είναι όσο το δυνατόν πιο απομακρυσμένο από τον πυρήνα (kernel) των Vista, ώστε να αυξάνεται η σταθερότητα τους. [38]

Εικόνα 9 Windows Vista Ultimate - Microsoft DirectX 10

Το αποτέλεσμα αυτών των αλλαγών είναι πολλές εφαρμογές που χρησιμοποιούσαν το DirectSound3D για τον τρισδιάστατο ήχο, να υπολειτουργούν ή να μην δουλεύουν καθόλου. Χαρακτηριστικό είναι ότι όσες εφαρμογές, (στην πλειοψηφία τους παιχνίδια, μιας και εκεί είναι αναγκαία η χρήση τρισδιάστατο ήχου), έκαναν χρήση του API DirectSound3D στο περιβάλλον των Vista, δεν υποστηρίζουν σε καμία περίπτωση των τρισδιάστατο ήχο, ενώ μιας και γίνεται προσομοίωση των εντολών του DirectSound και δεν επικοινωνεί πλέον απευθείας με τον επεξεργαστή της κάρτας ήχου, οι εφαρμογές γίνονται πιο αργές. [38,39]

Στα Windows XP μια εφαρμογή επικοινωνούσε σχετικά εύκολα με τον επεξεργαστή της κάρτας ήχου ανταλλάσσοντας δεδομένα με το DirectSound3D,το όποιο με τη σειρά του τα μετέφερε, μέσω των αντίστοιχων οδηγών, στην κάρτα ήχου. Με τον τρόπο αυτό μεταφέρονταν και οι σχετικές εντολές για τρισδιάστατους ήχους στην εκάστοτε εφαρμογή. [38]

Στα Windows Vista κάθε εφαρμογή που στέλνει εντολές σχετικές με ήχο αποτελεί από μόνη της ένα Audio session. Μέσα στα session οι εντολές ήχων ομαδοποιούνται σε ένα γκρουπ CPT (Cross Process Transport) και στέλνονται στο τμήμα του λειτουργικού, που είναι υπεύθυνο για την επεξεργασία τους .Τα εφέ επεξεργασίας στο τμήμα ήχου των Vista ονομάζονται APO (Audio Process Objects) και μπορεί να είναι, για παράδειγμα, ο καθορισμός της έντασης ή η μετατροπή της συχνότητας ενός ήχου. Προφανές είναι, ότι ταυτόχρονα μπορούν να εκτελούνται διάφορα session ήχων με παράλληλα CPT και

APO.Όμως όλα καταλήγουν σε ένα τελικό σημείο, το οποίο λέγεται KST (Kernel Streaming Transport) και με τη σειρά του μεταφέρει όλους τους ήχους στο λογισμικό–οδηγό της κάρτας ήχου του PC και από εκεί απευθείας στο σχετικό chip. Κάθε ηχητική επεξεργασία πραγματοποιείται από τα Windows και τον κεντρικό επεξεργαστή (CPU) του υπολογιστή και όχι από των επεξεργαστή της κάρτας ήχου. [38]

Εικόνα 10 Windows Vista - Vista vs XP sounds

Η Microsoft, παρ΄ όλο που άλλαξε πολλά πράγματα στο κομμάτι της ηχητικής επεξεργασίας των Vista, επέτρεψε τη χρήση εξωτερικών σετ εντολών και την απευθείας επικοινωνία τους με τον επεξεργαστή της εκάστοτε κάρτας ήχου. Τέτοια σετ ή πρότυπα εντολών είναι το ASIO[1]και η OpenAL. Με αποτέλεσμα το πολύ αναμενόμενο DirectX10 για τον ήχο να καθίσταται μη λειτουργικό και η OpenAL να αποτελεί το νέο standard. [38,40]

Η OpenAL και στα Vista δίνει την δυνατότητα στην εφαρμογή να επικοινωνεί απευθείας με το υλικό της κάρτας ήχου. Όσες εφαρμογές υποστηρίζουν την OpenAL μπορούν να εκμεταλλευτούν πλήρως τις δυνατότητες ενός επεξεργαστή τρισδιάστατων ήχων στα Vista. Αντίθετα οι εφαρμογές που βασίζονται στο DirectSound3D έχουν την δυνατότητα μέσω του προγράμματος με όνομα Alchemy της creative, να «μεταφράσουν» τις εντολές του DirectSound3D σε OpenAL και έτσι να ενεργοποιήσουν τον τρισδιάστατο ήχο στην εκάστοτε εφαρμογή. [38 ,39,41]

[1] ASIO -Audio Stream Input/Output: είναι ένα πρωτόκολλο υπολογιστών για τον ψηφιακό ήχο, που καθορίστηκε από την Steinberg. Παρέχει χαμηλό latency (χρόνο απόκρισης) και υψηλής ακρίβειας διεπαφή μεταξύ μιας εφαρμογής και κάρτας ήχου ενός υπολογιστή[40]

ΚΕΦΑΛΑΙΟ 2

ΤΟ AUDIO API OpenAL

Creative Labs Inc. OpenAL Version 1.1

1. ΓΕΝΙΚΑ ΣΤΟΙΧΕΙΑ

1.1 Ιστορική Αναδρομή

Οι πρώτες συζητήσεις για την δημιουργία της OpenAL ως API ήχου συμπληρωματικό στο OpenGL, άρχισαν περίπου το 1998. Υπήρξαν μερικές προσπάθειες για τη δημιουργία των αρχικών προδιαγραφών, αλλά αργότερα το 1999 η *Loki Entertainment Software*(http://www.lokigames.com/) είχε την ανάγκη για ένα API ακριβώς αυτού του τύπου, που να έχει προδιαγραφές και να υποστηρίζει εφαρμογές Linux. [35,42]

Εικόνα 1 Loki Entertainment Software

Περίπου σε εκείνο το χρονικό διάστημα, η *Loki* ξεκίνησε τις συζητήσεις με την *Creative Labs* για την τυποποίηση του API και την επέκταση των υποστηρίξιμων πλατφόρμων. Οι προδιαγραφές της *OpenAL 1.0* ανακοινώθηκαν στις αρχές του 2000 και η πρώτη έκδοση της OpenAL κυκλοφόρησε το ίδιο έτος για *Linux, MacOS 8/9, Windows* και *BeOS*. [35]

Μέσα στο 2000 η *Loki* δημιούργησε διάφορα παιχνίδια χρησιμοποιώντας την *OpenAL – Heavy Gear 2* και *Heretic 2Heavy 2* (και τα δύο για Linux). [35]

Το 2001,η *Creative Labs* κυκλοφόρησε την πρώτη *hardware-accelerated OpenAL* βιβλιοθήκη. Η βιβλιοθήκη αυτή υποστήριζε την SoundBlaster Live για MacOS 8/9 και Windows. [35]

Από το 2001, έχει υπάρξει συνεχής βελτίωση της OpenAL. Μερικές εκδόσεις για κάποιες πλατφόρμες κυκλοφορούν πιο σπάνια απ' ότι το 2000 (παραδείγματος χάριν *BeOS* και *MacOS 8/9*),αλλά έχουν προστεθεί περισσότερες πλατφόρμες (όπως *BSD, Solaris, IRIX, Mac OS X)* και διάσημες κονσόλες παιχνιδιών. Πλέον υποστηρίζει έναν μεγάλο αριθμό καρτών ήχου, ενώ υπάρχει πλήρης υποστήριξη για Creative και NVIDIA κάρτες ήχου σε περιβάλλον windows. [35]

1.2 Εφαρμογές που χρησιμοποιούν OpenAL

Η OpenAL έχει χρησιμοποιηθεί σε έναν μεγάλο αριθμό εφαρμογών, μέχρι σήμερα ,για διαφορετικές πλατφόρμες.[43,44]

GAMES	
• Alien Flux (Linux, Macintosh, Windows) • America's Army: Operations (Linux, Macintosh, Windows) • Armed Assault (Windows) • A Tale in the Desert II(Linux, Macintosh, Windows) • Battlefield 2(Windows) • Battlefield 2142(Windows) • Battle Just Started(Linux, Macintosh, Windows) • Blast Miner (Windows) • Bioshock (Windows) • Bridge Construction Set(Linux, Macintosh, Windows) • Call for Heroes: Pompolic Wars(Windows) • Call Of Juarez(Windows) • Cold War(Linux, Windows) • Colin McRae: DiRT(Windows) • D-Bug(Windows) • Dark Horizons: Lore(Linux, Macintosh, Windows) • Doom 3(Macintosh, Windows) • Dungeons & Dragons Online(Windows) • El Matador(Windows) • Escape From Monkey Island (Macintosh) • Eternal Lands(FreeBSD, Linux, Mac OS X, Windows) • E.V.E. Paradox(Linux, Windows) • FlightGear (FreeBSD, Linux, Macintosh, sgi, Solaris, Windows) • Ghost Recon: Advanced Warfighter (Windows) • Gish(Linux, Macintosh, Windows) • Glest (Linux Port) • Heavy Metal: F.A.K.K.^{2}(Linux) • Hot Potato Online(Linux, Windows) • Jedi Knight: Jedi Academy (Macintosh, Windows) • Jedi Knight 2 (Macintosh, Windows) • Just Cause(Windows) • Kohan(Linux) • Krabbit(Linux, XFree86) • Legends(FreeBSD, Linux, Macintosh, Windows, XFree86) • Lineage 2(Windows) • The Lord of the Rings Online: Shadows of Angmar (Windows) • Mage Knight: Apocalypse(Windows) • Marble Blast(Linux, Macintosh, Windows)	• Postal 2(Linux, Macintosh, Windows) • Prey(Windows) • Psychonauts(Windows) • Quake 4(Windows) • Regnum Online(Linux, Windows) • RocketBowl(Windows) • Rune(Linux) • Saints and Sinners Bowling(Windows) • Scorched 3D(FreeBSD, Linux, Mac OS X, Solaris, Windows) • Shellshock Nam '67(Windows) • Skyrocket Screensaver(Linux, Windows) • Slune(FreeBSD, Linux) • Soldier of Fortune(Linux) • Soldier of Fortune 2(Windows) • S.T.A.L.K.E.R.(Windows) • Star Trek Voyager: Elite Force{engine: ioquake3} (Linux , OS X, Windows) • Star Wars Republic Commando(Windows) • Stubbs the Zombie(Macintosh, Windows) • Sudden Strike 3: Arms For Victory(Windows) • SuperTux(Linux, Macintosh, Windows) • SuperTuxKart (Linux, Mac OS X, Windows) • SWAT 4(Windows) • Think Tanks(Linux, Macintosh, Windows) • Thunder&Lightning(Linux, Windows) • TORCS (FreeBSD, Linux, Mac OS X, Windows) • Tremulous{engine: ioquake3} (Linux , Windows) • Tribal Trouble(Linux, Macintosh, Windows) • Trigger(Linux, Macintosh, Windows) • Tribes 2(Linux) • Tribes: Vengeance(Windows) • UFO: Afterlight(Windows) • Ultimate Stunts(Linux, Windows) • Ultratron(Linux, Macintosh, Windows) • Unreal 2(Windows) • Unreal Tournament 2003(Linux, Macintosh, Windows) • Unreal Tournament 2004(Linux, Macintosh, Windows) • Urban Terror{engine: ioquake3} (Linux , OS X ?, Windows) • Vanguard: Saga of Heroes(Windows) • VDrift(FreeBSD, Linux, Mac OS X, Windows) • VegaStrike(Linux, Macintosh, Windows) • Void War(Windows)

• MegaCorps Online(Linux, Macintosh, Windows) • Minigolf Mania(Windows) • Minions of Mirth(Macintosh, Windows) • Myst Online: Uru Live(Windows) • Open Arena{engine: ioquake3} (Linux , OS X, Windows) • Orbz(Linux, Macintosh, Windows) • Penumbra: Overture(Linux, Windows) • PlaneShift(Linux, Macintosh, Windows)	• Warsow(Linux, Windows) • Wing Commander Saga(Macintosh, Windows) • World of Padman{engine: ioquake3} (Linux , OS X, Windows) • Wurm Online(Linux, Macintosh, Windows) • X^2: The Threat(Linux) • X^3: Reunion(Linux, XFree86) • X-Plane (Linux, Macintosh, Windows) • Zap! (Linux, Macintosh, Windows)

Engines και Libraries
• AgentAE(Linux, Macintosh, Windows) • C4 Engine(Macintosh, Windows) • Classical Moon Game Development Kit v18 (Chinese homepage)(Windows) • Classical Moon Game Development Kit v18 (English Intro)(Windows) • Decoy3D(Windows) • Delta3D(Linux, Windows) • Doom 3 Engine(Windows) • ELF Integrated Scripting System(Linux, Windows) • ioquake3 • Lightweight Java Game Library(Linux, Macintosh, Windows) • Luxinia(Windows) • PTK Game Engine(Macintosh, Windows) • SimGear - Simulator Construction Tools(FreeBSD, Linux, Macintosh, SGI, Windows) • Torque Engine(Linux, Macintosh, Windows) • Unreal Engine(Linux, Macintosh, Windows)

Εικόνα 2 Μερικοί από τους τίτλους που χρησιμοποιούν OpenAL

2. OPENAL AUDIO SYSTEM

2.1 Βασική δομή

Η OpenAL αποτελείται από τέσσερα βασικά API :

- *AL API -Audio Library*
- *ALC API - Audio Library Context*
- *EFX API - Effects Extension*
- *ALUT API - OpenAL Utility Toolkit*

Το *AL API* είναι ο πυρήνας του *OpenAL API*. Το API αυτό δεν έχει κάποιο προκαθορισμένο πλαίσιο λειτουργίας (context) και για αυτό τον λόγο χρησιμοποιεί το τρέχον, κάθε φορά, πλαίσιο, το οποίο προσδιορίζεται από το API ALC. Το ALC API είναι υπεύθυνο για τις λειτουργίες του context, καθώς και για την σύνδεση με το λειτουργικό σύστημα. Το EFX API παρέχει λειτουργίες effect με βάση το πρότυπο *EAX(Environmental Audio eXtensions)* της Creative Lab. Τέλος το ALUT API παρέχει την δυνατότητα εκτέλεσης μιας πιο ελαφριάς έκδοσης της OpenAL, πράγμα ιδιαίτερα κατάλληλο για κάποιον που θέλει να χρησιμοποιήσει εύκολα και γρήγορα το περιβάλλον αυτό. [35,36]

Ένα τυπικό πρόγραμμα OpenAL περιλαμβάνει τουλάχιστον δύο βασικές διαδικασίες:

- Η διαδικασία της αρχικοποίησης (Initializing) : Η διαδικασία αυτή ξεκινάει με τις κλήσεις που ανοίγουν μια συσκευή-device. Κατόπιν, ακολουθούν κλήσεις για να δημιουργήσουμε ένα περιβάλλον λειτουργίας, το οποίο καλείται *Context* AL και να το συνδέσουν με το device. Μόλις δημιουργηθεί ένα *Context* AL, ο προγραμματιστής είναι ελεύθερος να χρησιμοποιήσει τις εντολές του AL API.
- Η διαδικασία της εξόδου (Exiting): Αφορά το κλείσιμο του προγράμματος, όπου ο προγραμματιστής πρέπει να ακολουθήσει την αντίστροφη διαδικασία από αυτήν της αρχικοποίησης.

Οι διαδικασίες αυτές επιτυγχάνονται μέσω του ALC API. [35,36]

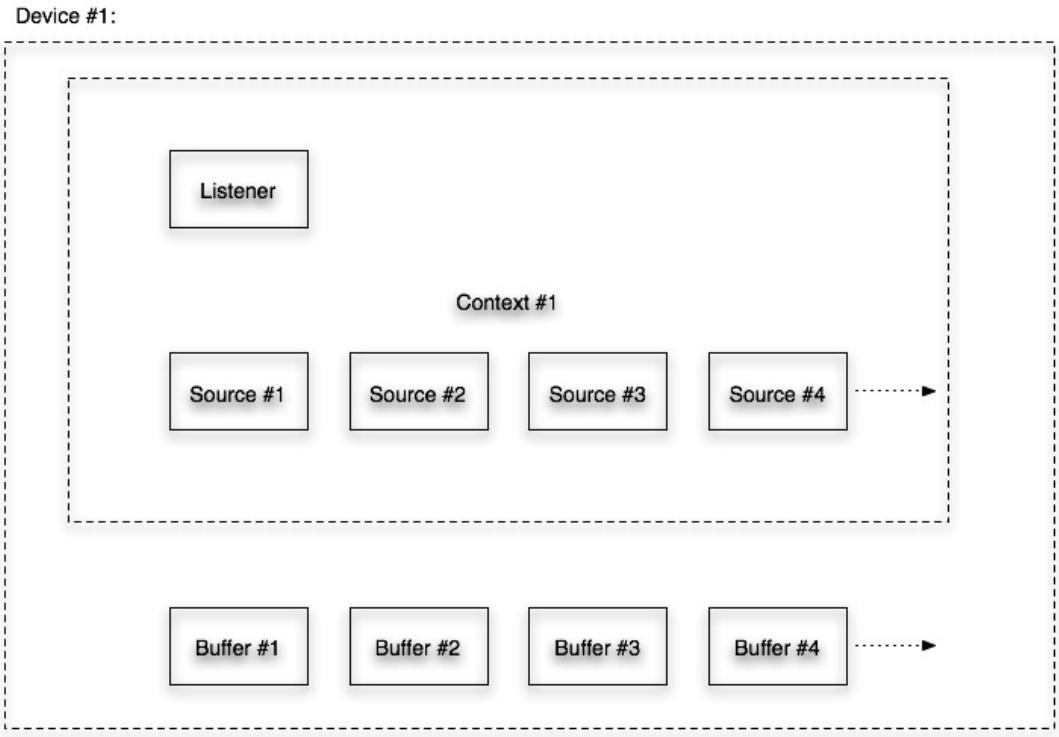
Η OpenAL διαθέτει τρία θεμελιώδη αντικείμενα ή objects: *buffers (προσωρινή αποθήκευση στοιχείων), sources (πηγές) και έναν listener (ακροατή)*. Ο καθορισμός ενός object δεν έχει επιπτώσεις στην ρύθμιση ενός άλλου. Κάθε ένα object μπορεί να τροποποιείται ανεξάρτητα από τα υπόλοιπα. Μπορούμε επίσης, να θέσουμε διάφορες καταστάσεις λειτουργίας (modes) σε κάθε ένα object, οι οποίες επηρεάζουν την τελική επεξεργασία, όπως για παράδειγμα τα μοντέλα απόστασης (Distance models). [35,36]

Τα αντικείμενα *buffers* χρησιμοποιούνται για την προσωρινή αποθήκευση αρχείων ήχου. Κατά την εκτέλεση του προγράμματος, τα αντικείμενα αυτά αρχικά φορτώνουν τα αρχεία ήχου στην μνήμη και όταν τους ζητηθεί ενώνονται με αντικείμενα source και τους παρέχουν τους ήχους. Συχνά τοποθετείται ένα μεγάλο σύνολο από *buffers* στην αρχή ενός προγράμματος ή σε διάφορα κρίσιμα σημεία που πρέπει να φορτωθούν στην εφαρμογή διάφορα αρχεία ήχου (π.χ. κατά την διάρκεια αλλαγής πίστας σε ένα παιχνίδι). [35,36]

Τα αντικείμενα *sources* αποθηκεύουν τις θέσεις, τις κατευθύνσεις και άλλες ιδιότητες των objects στον τρισδιάστατο εικονικό χώρο. Ένα αντικείμενο *buffer* συνδέεται μαζί τους, ώστε να τους παρέχει τον ήχο που πρέπει να ρυθμίσουν, σύμφωνα με τις ιδιότητες που έχουν αποθηκεύσει. Τα αντικείμενα *sources* υποβάλλονται σε επεξεργασία ανεξάρτητα το ένα από το άλλο. [35,36]

Σε κάθε *Context* υπάρχει μόνο ένας *listener*. Οι ιδιότητες του *listener* είναι παρόμοιες με τις ιδιότητες των *sources*, αλλά αντιπροσωπεύουν τον τελικό ήχο που θα ακούσει ο χρήστης της εφαρμογής. Όλα τα *sources* τοποθετούνται (μιξάρονται και εκτελούνται) σχετικά με τον *listener*. [35,36]

Το αποτέλεσμα αυτής της βασικής δομής είναι να παρέχει την δυνατότητα δημιουργίας πολλών και διαφορετικών *sources, buffers* και ενός ενιαίου *listener,* έτσι ώστε να επιτυγχάνεται μια συνεχής ενημέρωση της θέσης και του προσανατολισμού των *sources* και *listener,* με αποτέλεσμα ένα δυναμικά εναλλασσόμενο τρισδιάστατο ακουστικό κόσμο. [35,36]



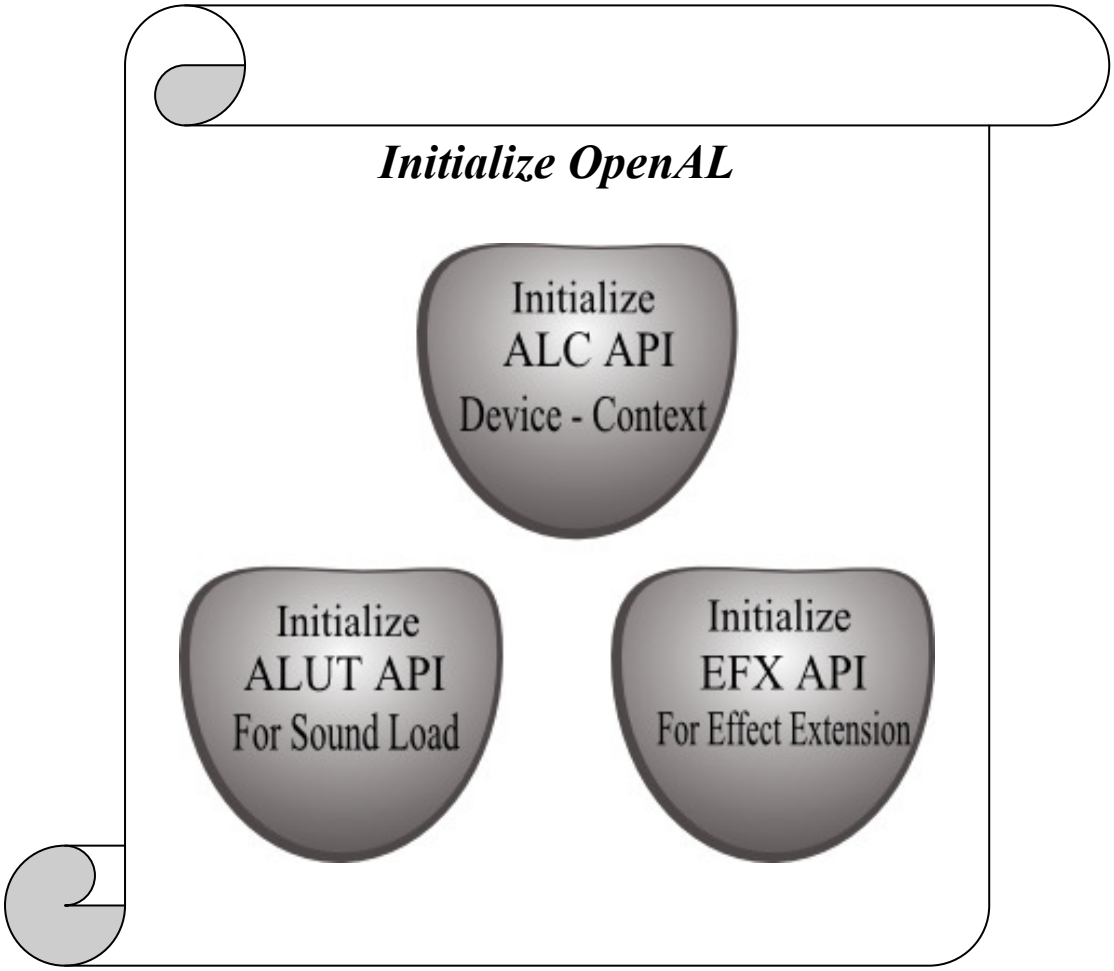
Εικόνα 4 Διάγραμμα που παρουσιάζει τα θεμελιώδη objects της OpenAL και τις σχέσεις τους με το context και device

Συνοψίζοντας: Κατά την διαδικασία της αρχικοποίησης της *OpenAL*, τουλάχιστον ένα device πρέπει να ανοίξει. Μέσα σε αυτό το *device* τουλάχιστον ένα context πρέπει να δημιουργηθεί. Και τέλος μέσα σε αυτό το *context* μόνο ένα αντικείμενο *listener* μπορεί να δημιουργηθεί, ενώ ταυτόχρονα μπορεί να δημιουργηθεί ένα πλήθος από αντικείμενα *sources*. Σε κάθε source μπορεί να συνδεθεί ένα ή περισσότερα αντικείμενα *buffers*. Τα *buffers* δεν είναι μέρος ενός συγκεκριμένου *context,* αλλά μοιράζονται μεταξύ όλων των *contexts* σε ένα *device.* [35,36]

2.2 Βασική δομή προγράμματος

Η βασική δομή ενός προγράμματος OpenAL χωρίζεται σε τρία βασικά μέρη: *Initializing(Αρχικοποίηση), Processing Loop (Επεξεργασία)* και *exiting (Έξοδος).* [35,36]

Κατά την *αρχικοποίηση* ενός προγράμματος ορίζονται devices και contexts που θα χρησιμοποιηθούν. Επίσης σε αυτό το στάδιο αρχικοποιούνται και αλλά APIs, ώστε να μπορούν να χρησιμοποιηθούν αργότερα. [35,36]

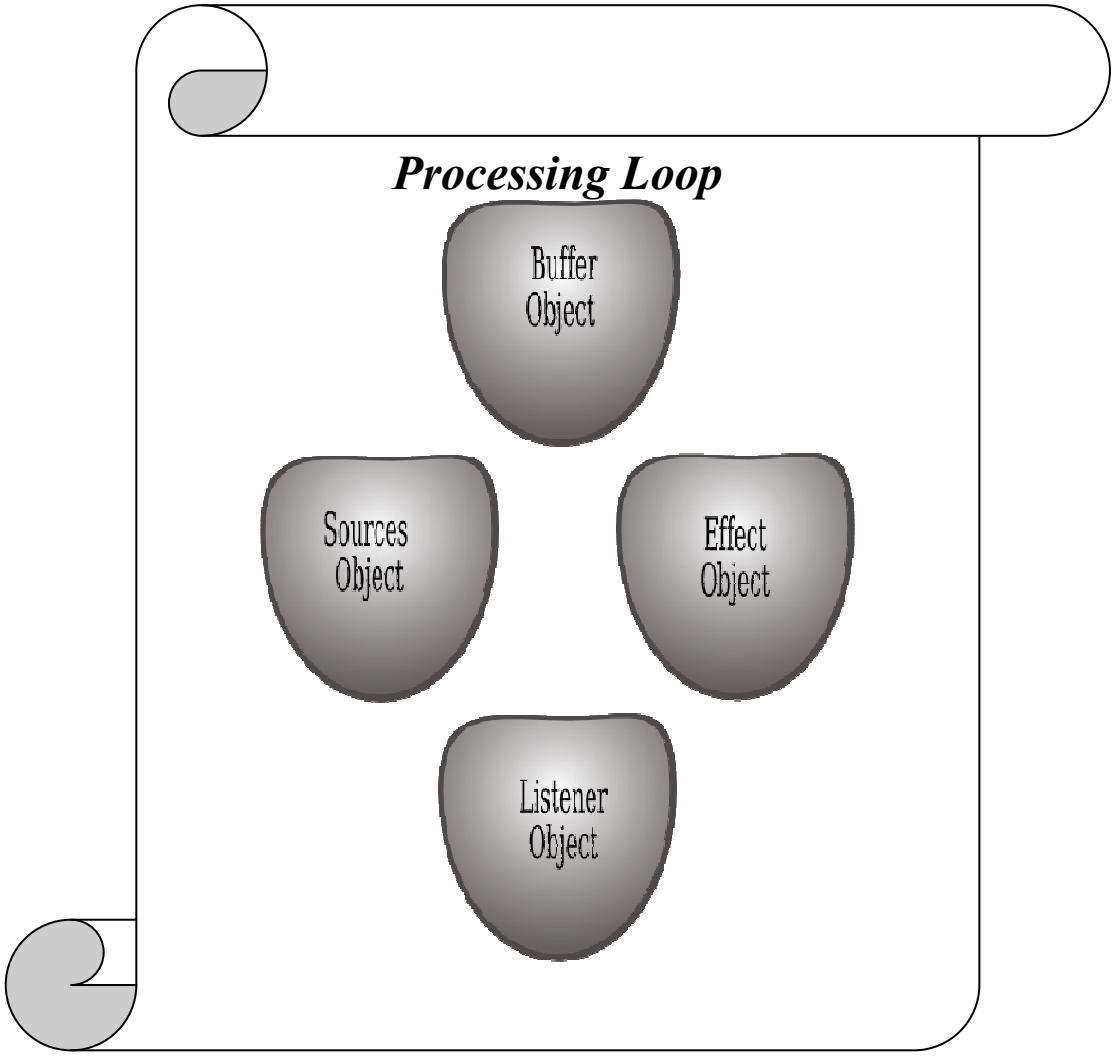


Εικόνα 5 Initialize OpenAL

Η αρχικοποίηση ενός προγράμματος *ξεκινάει πάντα* με εντολές από το *ALC API.* Μέσω των εντολών αυτών ορίζονται το *device* και τα *contexts* που θα χρησιμοποιήσει το τελικό πρόγραμμα. Αυτή η διαδικασία είναι απαραίτητη για οποιοδήποτε πρόγραμμα θέλει να χρησιμοποιήσει το OpenAL API. [35,36]

Επίσης κατά την αρχικοποίηση ορίζονται και αρχικοποιούνται διάφορα άλλα APIs, που δεν είναι όμως απαραίτητα για την λειτουργία του *AL API*, όπως το *EFX API,* το οποίο είναι υπεύθυνο για την λειτουργία διάφορων effects και παρέχεται μαζί με το OpenAL API, καθώς και το *ALUT API,* το οποίο αυτή την στιγμή παρέχεται ξεχωριστά από το OpenAL API, αλλά χρησιμοποιείται συχνά για την φόρτωση δεδομένων ήχου σε buffers της AL. [35,36,45]

Στο τμήμα που ονομάζεται *επεξεργασία,* ο προγραμματιστής μπορεί να χρησιμοποιήσει όλα τα objects που διαθέτει το OpenAL API για να δημιουργήσει το επιθυμητό αποτέλεσμα, όπως επίσης και όλα τα objects των άλλων παρεχόμενων APIs (όπως ALUT και EFX). [35,36,45]

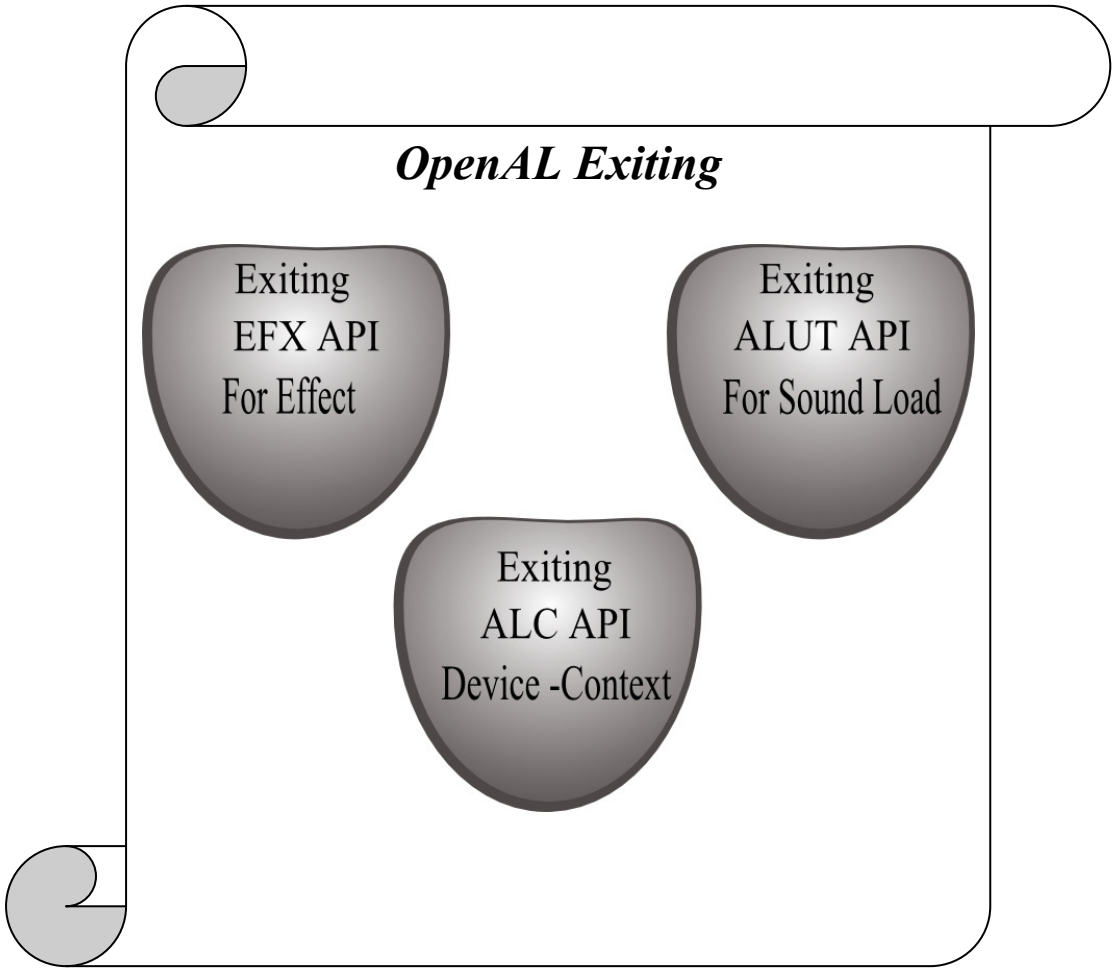


Εικόνα 6 Processing Loop

Πρέπει να σημειωθεί ότι στο ξεκίνημα της *επεξεργασίας* δημιουργούνται διάφορα objects όπως buffers, sources, effects και ένας listener, ενώ κατά το κλείσιμο του όλα αυτά τα objects διαγράφονται με αντίστοιχες εντολές πριν το τμήμα εξόδου. [35,36]

Το τελευταίο τμήμα ενός OpenAL προγράμματος είναι η *έξοδος* που αποτελεί *ακριβώς την αντίστροφη διαδικασία* από την αρχικοποίηση. Ακόμα και σε επίπεδο εντολών ακολουθείται ακριβώς η αντίστροφη διαδικασία (δηλ. το device που ενεργοποιείται πρώτο κατά την αρχικοποίηση, τώρα είναι αυτό που απενεργοποιείται τελευταίο). [35,36]

Έτσι αρχικά κλείνουν τα APIs EFX και ALUT, εφόσον έχουν χρησιμοποιηθεί. Ακολουθεί η διαγραφή των contexts και τελευταίο κλίνει το device, μέσω των εντολών του ALC API. Μετά τη διαδικασία αυτή, είναι προφανές ότι κανένα object δεν μπορεί να χρησιμοποιηθεί. Από την στιγμή που διαγράφηκε το context και έκλεισε το device, για να επαναχρησιμοποιηθεί η OpenAL πρέπει να γίνει ξανά *αρχικοποίηση*. [35,36,45]



Εικόνα 7 OpenAL Exiting

2.3 Τύποι δεδομένων

Το API OpenAL διαθέτει τους δικούς του τύπους δεδομένων και μεταβλητών. Γενικά οι τύποι δεδομένων είναι ευέλικτοι, δηλαδή μπορούν να χρησιμοποιηθούν και για το AL και για το ALC (π.χ. οι τύποι ALCvoid και ALvoid μπορούν να χρησιμοποιηθούν ανεξαρτήτως των εντολών που περιέχουν, ενώ επίσης μπορεί να χρησιμοποιηθεί και η γνωστή void). Σε ορισμένες περιπτώσεις όμως, οι εντολές αφορούν τους εξειδικευμένους τύπους μεταβλητών με βάση το API από το οποίο προέρχονται. [35,36]

Ανεξαρτήτως των απαιτήσεων του API, θεωρείται γενικά σωστό η μορφοποίηση του προγράμματος να ακολουθεί την εξειδικευμένη μορφή, ακόμα και αν δεν είναι αναγκαίο από το API, ώστε ο κώδικας να είναι πιο εύκολα αναγνώσιμος κατά την διόρθωση και συμπλήρωση λειτουργιών. [35,36]

ALC Primitive Data Types	
ALCdevice	Define values για τον ορισμό των device
ALCcontext	Define values για τον ορισμό των context
ALCboolean	8-bit boolean
ALCchar	Χαρακτήρας
ALCbyte	Προσημασμένος 8-bit ακέραιος αριθμός συμπληρώματος ως προς 2
ALCubyte	Μη προσημασμένος 8-bit ακέραιος αριθμός
ALCshort	Προσημασμένος 16-bit ακέραιος αριθμός συμπληρώματος ως προς 2
ALCushort	Μη προσημασμένος 16-bit ακέραιος αριθμός
ALCint	Προσημασμένος 32-bit ακέραιος αριθμός συμπληρώματος ως προς 2
ALCuint	Μη προσημασμένος 32-bit ακέραιος αριθμός
ALCsizei	Μη αρνητικό 32-bit δυαδικό μέγεθος ακέραιων αριθμών
ALCenum	Enumerated 32-bit τιμή
ALCfloat	32-bit IEEE754 μεγάλη πραγματική τιμή
ALCdouble	64-bit IEEE754 πολύ μεγάλη πραγματική τιμή
ALCvoid	ALC void τύπος
AL Primitive Data Types	
ALboolean	8-bit boolean
ALchar	Χαρακτήρας
ALbyte	Προσημασμένος 8-bit ακέραιος αριθμός συμπληρώματος ως προς 2
ALubyte	Μη προσημασμένος 8-bit ακέραιος αριθμός
ALshort	Προσημασμένος 16-bit ακέραιος αριθμός συμπληρώματος ως προς 2
ALushort	Μη προσημασμένος 16-bit ακέραιος αριθμός
ALint	Προσημασμένος 32-bit ακέραιος αριθμός συμπληρώματος ως προς 2
ALuint	Μη προσημασμένος 32-bit ακέραιος αριθμός
ALsizei	Μη αρνητικό 32-bit δυαδικό μέγεθος ακέραιων αριθμών
ALenum	Enumerated 32-bit τιμή
ALfloat	32-bit IEEE754 μεγάλη πραγματική τιμή (floating-point)
ALdouble	64-bit IEEE754 πολύ μεγάλη πραγματική τιμή (floating-point)
ALvoid	ALC void τύπος

2.4 Error type

Η OpenAL διαθέτει επίσης και ένα δικό της σύστημα ευρέσεως σφαλμάτων. Είναι πολύ σημαντικό ένας προγραμματιστής OpenAL να γνωρίζει να χρησιμοποιεί σωστά την λειτουργία εύρεσης σφαλμάτων και να αποκωδικοποιεί τα μηνύματα. [35,36]

Η OpenAL ανιχνεύει μόνο ένα υποσύνολο εκείνων των όρων που θα μπορούσαν να θεωρηθούν ως σφάλματα. Αυτό συμβαίνει επειδή σε πολλές περιπτώσεις ο έλεγχος σφαλμάτων θα επιβάρυνε την εκτέλεση ενός προγράμματος κατά το run-time. [35,36]

Τα APIs ALC και AL διαθέτουν τους δικούς τους τύπους σφαλμάτων, όπως φαίνεται στους παρακάτω πίνακες:

ALC ERROR	
ALC_NO_ERROR	Δεν υπάρχει κανένα ALC τρέχον λάθος
ALC_INVALID_DEVICE	Το device handle ή το όνομα του διευκρινισμένου driver / server είναι άκυρο.(No device)
ALC_INVALID_CONTEXT	Το όρισμα του context δεν είναι μια έγκυρη ταυτότητα context (Invalid context ID)
ALC_INVALID_ENUM	Ένα σύμβολο που χρησιμοποιείται δεν είναι έγκυρο ή είναι μη εφαρμόσιμο. Άκυρη παράμετρος (bad enum)
ALC_INVALID_VALUE	Μια τιμή δεν είναι έγκυρη ή είναι μη εφαρμόσιμη. Άκυρη αξία παραμέτρου enum.(bad enum)
ALC_OUT_OF_MEMORY	Η μνήμη δεν επαρκεί (Out of memory)

AL ERROR	
AL_NO_ERROR	Δεν υπάρχει κανένα AL τρέχον λάθος
AL_INVALID_NAME	Άκυρο όνομα παραμέτρου
AL_ILLEGAL_ENUM	Άκυρη παράμετρος.
AL_INVALID_ENUM	Ένα σύμβολο που χρησιμοποιείται δεν είναι έγκυρο ή είναι μη εφαρμόσιμο. Άκυρη παράμετρος(bad enum)
AL_INVALID_VALUE	Μια αξία δεν είναι έγκυρη ή είναι μη εφαρμόσιμη. Άκυρη αξία παραμέτρου enum.(bad enum)
AL_ILLEGAL_COMMAND	Εσφαλμένη κλήση
AL_INVALID_OPERATION	Εσφαλμένη κλήση
AL_OUT_OF_MEMORY	Η μνήμη δεν επαρκεί (Out of memory)

Προφανές είναι, ότι τα ALC Errors αναφέρονται και χρησιμοποιούνται στο τμήμα *αρχικοποίησης και εξόδου* ενός προγράμματος, ενώ τα AL Errors χρησιμοποιούνται στο *τμήμα επεξεργασίας* και είναι ίδια και για το *EFX API,* που δεν διαθέτει δικά του *Errors.* [35,36]

Το ALUT API, με την σειρά του, διαθέτει ένα πιο μεγάλο εύρος ελέγχου σφαλμάτων, τα οποία εκτός από τα εξειδικευμένα ALUT σφάλματα, περιλαμβάνουν και σφάλματα των AL και ALC APIs. Ο πλήρης κατάλογος ALUT σφαλμάτων παρουσιάζεται στον παρακάτω πίνακα: [35,36,45]

ALUT ERROR	
ALUT_ERROR_NO_ERROR	Κανένα ALUT λάθος δεν βρέθηκε
ALUT_ERROR_OUT_OF_MEMORY	Η μνήμη δεν επαρκεί.
ALUT_ERROR_INVALID_ENUM	Δόθηκε μια άκυρη τιμή enum. Άκυρη παράμετρος (bad enum).
ALUT_ERROR_INVALID_VALUE	Δόθηκε μια άκυρη τιμή .
ALUT_ERROR_INVALID_OPERATION	Η λειτουργία είναι άκυρη στην επικρατούσα κατάσταση ALUT.
ALUT_ERROR_NO_CURRENT_CONTEXT	Δεν υπάρχει κανένα τρέχον AL context.
ALUT_ERROR_AL_ERROR_ON_ENTRY	Υπήρξε ήδη ένα λάθος AL κατά την εισαγωγή σε μια λειτουργία ALUT.
ALUT_ERROR_ALC_ERROR_ON_ENTRY	Υπήρξε ήδη ένα λάθος ALC κατά την εισαγωγή σε μια λειτουργία ALUT.
ALUT_ERROR_OPEN_DEVICE	Υπήρξε ένα λάθος κατά το άνοιγμα ενός ALC device
ALUT_ERROR_CLOSE_DEVICE	Υπήρξε ένα λάθος κατά το κλείσιμο ενός ALC device
ALUT_ERROR_CREATE_CONTEXT	Υπήρξε ένα λάθος κατά την δημιουργία ενός ALC context .
ALUT_ERROR_MAKE_CONTEXT_CURRENT	Δεν μπορεί να τεθεί ένα context ως το τρέχον ALC context .
ALUT_ERROR_DESTROY_CONTEXT	Υπήρξε ένα λάθος κατά την καταστροφή του ALC context
ALUT_ERROR_GEN_BUFFERS	Υπήρξε ένα λάθος στην δημιουργία ενός AL buffer
ALUT_ERROR_BUFFER_DATA	Υπήρξε ένα λάθος κατά την διάρκεια εισαγωγής στοιχείων στον buffer AL
ALUT_ERROR_IO_ERROR	I/O error. Συμβουλευτείτε το errno για περισσότερες λεπτομέρειες
ALUT_ERROR_UNSUPPORTED_FILE_TYPE	Μη υποστηρίξιμος τύπος αρχείου.
ALUT_ERROR_UNSUPPORTED_FILE_SUBTYPE	Μη υποστηρίξιμος επιλογή εντός μιας υποκατηγορίας ενός ήδη χρησιμοποιημένου τύπου αρχείου
ALUT_ERROR_CORRUPT_OR_TRUNCATED_DATA	Το αρχείο ήχου έχει αλλοιωθεί ή έχει περικοπεί

Και τα τρία APIs χρησιμοποιούν την ίδια ακριβώς μεθοδολογία εύρεσης σφαλμάτων. Οι συναρτήσεις για τα AL, ALC και ALUT αντίστοιχα είναι :

```
ALenum      alGetError   (ALvoid);
ALCenum     alcGetError (ALCdevice *device);
ALenum      alutGetError (void);
```

Και οι τρεις συναρτήσεις ανακτάνε και εκκαθαρίζουν την επικρατούσα κατάσταση λάθους για το αντίστοιχο API. Επίσης, όπως φαίνεται και από την σύνταξη τους, το αποτέλεσμα τους είναι ένα αριθμός ALenum και ALCenum, που αντιστοιχεί σε ένα #define της βιβλιοθήκης (π.χ. #define AL_NO_ERROR 0) . [35,36,45]

Ιδιαίτερη σημασία στις Get Error συναρτήσεις έχουν τα εξής : [35,36,45]

- Ανιχνεύουν μόνο ένα υποσύνολο σφαλμάτων.
- Χρησιμοποιούνται μετά το άνοιγμα ενός device και την δημιουργία ενός context. Αν χρησιμοποιηθούν πριν τη χρήση ενός device και την δημιουργία ενός προκαθορισμένου context, είναι δεδομένο ότι θα προκύψει ένα Error.[1]
- Πρέπει να εκκαθαρίζονται πριν χρησιμοποιηθούν για πρώτη φορά σε ένα τμήμα του προγράμματος, ώστε να είναι σίγουρο ότι δεν έχουν κρατήσει κάποιο παλιότερο λάθος. Η εκκαθάριση μιας Get Error συνάρτησης γίνεται καλώντας την σε ένα σημείο του κώδικα, όπου δεν υπάρχει σίγουρα λάθος AL, όπως φαίνεται στο παράδειγμα που ακολουθεί (Παράδειγμα 1).
- Οι συναρτήσεις Get Error αποθηκεύουν μόνο την τρέχουσα κατάσταση σφάλματος που παρουσιάσθηκε στην πιο πρόσφατη εντολή που χρησιμοποιήθηκε πριν από αυτές.
- Μπορούν να χρησιμοποιηθούν μέσα σε λογικές πράξεις, όπως οποιαδήποτε συνάρτηση της C (π.χ. if (AL_NO_ERROR == alGetError ( ))).
- Η έξοδος τους είναι πάντα ένας αριθμός που υποδηλώνει την κατάσταση σφάλματος που αναλύεται στους πίνακες. Η εκτύπωση του τρέχοντος σφάλματος στην οθόνη σε string, είναι μια άλλη διαδικασία εντελώς ξεχωριστή και μπορεί να επιτευχθεί μέσω της συναρτήσεως *GetString,* που θα αναλυθεί παρακάτω.

Εικόνα 7 Γελοιογραφία : Όταν δεν μπορούμε να κατανοήσουμε τα μηνύματα σφαλμάτων δεν μπορούμε να κατανοήσουμε την ίδια την λειτουργία του προγράμματος και του υπολογιστή.

[1] Περισσότερες πληροφορίες για την αρχικοποίηση σε σχέση με τα σφάλματα, δίνονται στο επόμενο υποκεφάλαιο.

Στο παράδειγμα 1 μπορούμε να δούμε πως ακριβώς χρησιμοποιούνται οι συναρτήσεις Get Error, ώστε να έχουμε το επιθυμητό αποτέλεσμα:

Παράδειγμα 1

```
/*Ορισμένες τυπικές κλίσεις*/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <Windows.h>
/*Κλίση της OpenAL*/
#include <al.h>
#include <alc.h>
#include <alut.h>
#include <efx.h>
#include "myinclude.h"
int main()
{

/*1 ΜΕΡΟΣ*/
ALenum Error; /* Αποθήκευση του error status*/
ALuint  Buffer; /*Το όνομα ενός Buffer */
ALuint  Sources; /*Το όνομα ενός Source */
ALuint  Sources2; /*Το όνομα ενός Source */

InitOpenAL(); /*Συνάρτηση του προγραμματιστή που κάνει initialize την OpenAL*/
InitALUT(); /*Συνάρτηση του προγραμματιστή που κάνει initialize την OpenAL ALUT*/

/*2 ΜΕΡΟΣ*/
alGetError(); /*Εκκαθαρίζει την επικρατούσα κατάσταση λάθους AL*/
alutGetError();/*Εκκαθαρίζει την επικρατούσα κατάσταση λάθους ALUT*/


/*3 ΜΕΡΟΣ*/
Buffer = alutCreateBufferFromFile (test.wav);
/*Φορτώνει ένα αρχείο ήχου μέσω του ALUT API σε ένα buffer*/

 if (Buffer == AL_NONE)
  {/*Ελέγχει αν έχει φορτωθεί κάτι στον buffer*/

    Error = alutGetError();
  /*Αν δεν έχει φορτωθεί κάτι, γίνεται έλεγχος για σφάλματα*/

  printf ( "\n Error loading: %s\n", alutGetErrorString (Error));
  /*Εκτυπώνει το τρέχον Error ALUT */
 }

/*4 ΜΕΡΟΣ*/
alGenSources (1, &Sources); /*Δημιουργεί ένα source */
Error = alGetError (); /*Γίνεται έλεγχος για σφάλματα*/
if ( Error != AL_NO_ERROR ) /*Έλεγχος για το αν βρέθηκε ένα τρέχον error*/
 {
  printf ("!!Error Generate Source: %s\n", alGetString (Error));
  /*Εκτυπώνει το τρέχον Error AL*/
 }
/*Program exiting- END*/
}
```

Το παράδειγμα 1 χωρίζεται σε 4 μέρη. Στο *1ο μέρος* γίνεται η δήλωση των μεταβλητών που θα χρησιμοποιηθούν από το πρόγραμμα. Η μεταβλητή που μας ενδιαφέρει στην προκείμενη περίπτωση και πρέπει να εξετάσουμε πιο προσεχτικά είναι :

```
ALenum Error; /* Αποθήκευση του error status*/
```

Σε αυτήν την μεταβλητή θα αποθηκεύεται κάθε φορά το error status του προγράμματος, όποιο και αν είναι αυτό. Μετά την δήλωση των μεταβλητών γίνεται η αρχικοποίηση της AL και ALUT μέσω των συναρτήσεων που έχει ήδη φτιάξει ένας προγραμματιστής και θα αναλύσουμε σε επόμενα κεφαλαία. [35,36,45]

Στο *2ο μέρος* του παραδείγματος γίνεται εκκαθάριση της επικρατούσας κατάστασης λάθους για al και alut error:

```
alGetError();  /*Εκκαθαρίζει την επικρατούσα κατάσταση λάθους AL*/
alutGetError();/*Εκκαθαρίζει την επικρατούσα κατάσταση λάθους ALUT*/
```

Η εκκαθάριση σφαλμάτων είναι μια γενική μεθοδολογία που χρησιμοποιείται από τους προγραμματιστές. Οι Get Error συναρτήσεις κάθε φορά που εκτελούνται από ένα πρόγραμμα συγκρατούν την επικρατούσα κατάσταση λάθους. Αυτό έχει ως αποτέλεσμα να εκκαθαρίζουν (διαγράφουν) την προηγούμενη κατάσταση λάθους. Έτσι τρέχοντας μια φορά την συνάρτηση επιτυγχάνεται ο εκκαθαρισμός της από τυχόν προηγούμενα σφάλματα, που μπορεί να έχει συγκρατήσει και εφόσον τρέξει σε ένα σημείο του προγράμματος που δεν έχει σφάλμα, επιτυγχάνεται ο πλήρης εκκαθαρισμός της, δηλαδή η συγκράτηση του status : AL_NO_ERROR για την alGetError, ALUT_ERROR_NO_ERROR για την alutGetError και ALC_NO_ERROR για την alcGetError, που ουσιαστικά σημαίνει ότι δεν είναι ανιχνεύσιμο κανένα λάθος. [35,36,45]

Στα μέρη 3ο και 4ο θα εξετάσουμε μόνο τα σημεία που έχουν σχέση με τον τρόπο που χειριζόμαστε τα σφάλματα και όχι τις υπόλοιπες συναρτήσεις που χρησιμοποιούνται ως παράδειγμα και οι οποίες θα αναλυθούν σε ακόλουθα κεφάλαια.

Στο *3ο Μέρος* αρχικά χρησιμοποιείται μια συνάρτηση ALUT που εκτελεί μια οποιαδήποτε λειτουργία

```
Buffer = alutCreateBufferFromFile (test.wav);
/*Φορτώνει ένα αρχείο ήχου μέσω του ALUT API σε ένα buffer*/
```

Στην συνεχεία ελέγχεται αν όντως η συγκεκριμένη λειτουργία έχει επιτευχθεί.

```
if (Buffer == AL_NONE) /*Ελέγχει αν έχει φορτωθεί κάτι στον buffer*/
```

Αν η if είναι αληθής, τότε δεν έχει επιτευχθεί σωστά η λειτουργία και επομένως χρειάζεται έλεγχος για σφάλματα Όπως φαίνεται παρακάτω, εκτελείται η alutGetError και αποθηκεύεται το Error status στην μεταβλητή που ορίσαμε στην αρχή του προγράμματος: [35,36,45]

```
    Error = alutGetError();
  /*Αν δεν έχει φορτωθεί κάτι, γίνεται έλεγχος για σφάλματα*/
```

Τέλος τοποθετούμε την μεταβλητή που περιέχει τον αριθμό του σφάλματος στην alutGetErrorString, ώστε να εκτυπωθεί το error που δημιουργήθηκε, εφόσον βέβαια αυτό το σφάλμα μπορεί να αναγνωστεί από την alutGetError:

```
  printf ( "\n Error loading: %s\n", alutGetErrorString (Error));
  /*Εκτυπώνει το τρέχον Error ALUT */
```

Όπως προαναφέραμε, οι συναρτήσεις GetString είναι αυτές που δίνουν την δυνατότητα εκτύπωσης και ερμηνείας των σφαλμάτων που δημιουργούνται. Η ALUT χρησιμοποιεί την συνάρτηση alutGetErrorString, όπου ουσιαστικά κάνει μια παρά πολύ απλή λειτουργία, δηλαδή μετατρέπει τα errors status σε κατανοητά για τον άνθρωπο μηνύματα. [35,36,45]

Η συνάρτηση συντάσσεται ως εξής:

```
const char *alutGetErrorString (ALenum error);
```

Ο πηγαίος κώδικας της συνάρτησης που δείχνει και τα μηνύματα που εκτυπώνονται εντέλει στην οθόνη του χρήστη φαίνεται παρακάτω: [35,36,45]

```
  const char *alutGetErrorString (ALenum error)
{
  switch (error)
    {
    case ALUT_ERROR_NO_ERROR:
      return "No ALUT error found";

    case ALUT_ERROR_OUT_OF_MEMORY:
      return "ALUT ran out of memory";

    case ALUT_ERROR_INVALID_ENUM:
      return "ALUT was given an invalid enumeration token";

    case ALUT_ERROR_INVALID_VALUE:
      return "ALUT was given an invalid value";

    case ALUT_ERROR_INVALID_OPERATION:
      return "The operation was invalid in the current ALUT state";

    case ALUT_ERROR_NO_CURRENT_CONTEXT:
      return "There is no current AL context";

    case ALUT_ERROR_AL_ERROR_ON_ENTRY:
      return "There was already an AL error on entry to an ALUT function";

    case ALUT_ERROR_ALC_ERROR_ON_ENTRY:
      return "There was already an ALC error on entry to an ALUT function";

    case ALUT_ERROR_OPEN_DEVICE:
      return "There was an error opening the ALC device";

    case ALUT_ERROR_CLOSE_DEVICE:
      return "There was an error closing the ALC device";

    case ALUT_ERROR_CREATE_CONTEXT:
      return "There was an error creating an ALC context";

    case ALUT_ERROR_MAKE_CONTEXT_CURRENT:
      return "Could not change the current ALC context";
```

```
    case ALUT_ERROR_DESTROY_CONTEXT:
      return "There was an error destroying the ALC context";

    case ALUT_ERROR_GEN_BUFFERS:
      return "There was an error generating an AL buffer";

    case ALUT_ERROR_BUFFER_DATA:
      return "There was an error passing buffer data to AL";

    case ALUT_ERROR_IO_ERROR:
      return "I/O error";

    case ALUT_ERROR_UNSUPPORTED_FILE_TYPE:
      return "Unsupported file type";

    case ALUT_ERROR_UNSUPPORTED_FILE_SUBTYPE:
      return "Unsupported mode within an otherwise usable file type";

    case ALUT_ERROR_CORRUPT_OR_TRUNCATED_DATA:
      return "The sound data was corrupt or truncated";

    default:
      return "An impossible ALUT error condition was reported?!?";
    }
}
```

Επιστρέφοντας στο παράδειγμα 1 και εξετάζοντας αυτήν την φορά το *Μέρος 4ο* είναι φανερό ότι η διαδικασία που ακολουθείται είναι παρόμοια, αλλά αυτήν την φορά αφορά το AL API. [35,36,45]

Αρχικά χρησιμοποιείται μια συνάρτηση AL που εκτελεί μια λειτουργία

```
  alGenSources (1, &Sources); /*Δημιουργεί ένα source */
```

Στην συνεχεία γίνεται έλεγχος για AL σφάλματα :

```
  Error = alGetError (); /*Γίνεται έλεγχος για σφάλματα*/
```

Αν το error status που αποθηκεύτηκε στην μεταβλητή error δεν είναι AL_NO_ERROR,τότε προφανώς έχει προκύψει ένα λάθος που εκτυπώνεται στην οθόνη μέσω της εντολής alGetString. [35,36,45]

```
  if ( Error != AL_NO_ERROR )
{/*Έλεγχος για το αν βρέθηκε ένα τρέχον error*/
  printf ("!!Error Generate Source: %s\n", alGetString (Error));
  /*Εκτυπώνει το τρέχον Error AL*/
```

Όπως βλέπουμε η συνάρτηση alGetString είναι αυτή που χρησιμοποιείται για την εκτύπωση των σφαλμάτων. Όμως έχει μια βασική διαφορά από την συνάρτηση alutGetErrorString. Η alGetString[1] όπως και η alcGetString[2] (για το ALC API) δεν χρησιμοποιούνται μόνο για τα string των σφαλμάτων, αλλά και για άλλες λειτουργίες. [35,36,45]

[1] Η συνάρτηση αναλύεται περαιτέρω στο υποκεφάλαιο που αναφέρεται το AL API.
[2] Η συνάρτηση αναλύεται περαιτέρω στο υποκεφάλαιο που αναφέρεται το ALC API.

AL string Error Messages /alGetString	
AL_NO_ERROR	No Error
AL_INVALID_NAME	Invalid Name
AL_ILLEGAL_ENUM	Illegal Enum
AL_INVALID_ENUM	Invalid Enum.
AL_INVALID_VALUE	Invalid Value
AL_ILLEGAL_COMMAND	Illegal command
AL_INVALID_OPERATION	Invalid Operation
AL_OUT_OF_MEMORY	Out of Memory

ALC string Error Messages /alcGetString	
ALC_NO_ERROR	No Error
ALC_INVALID_DEVICE	Invalid Device
ALC_INVALID_CONTEXT	Invalid Context
ALC_INVALID_ENUM	Invalid Enum
ALC_INVALID_VALUE	Invalud Value
ALC_OUT_OF_MEMORY	Out of Memory

Πρέπει να σημειωθεί ότι η συνάρτηση alcGetError λειτουργεί ακριβώς με τον ίδιο τρόπο που λειτουργεί και η alGetError. Επειδή όμως αναφέρεται σε ένα συγκεκριμένο device θα αναλυθεί περαιτέρω στο κεφαλαίο του ALC API. [35,36,45]

Μια τελευταία ενδιαφέρουσα γενική παρατήρηση για τα σφάλματα που φαίνονται στο 3^{ο} μέρος του παραδείγματος είναι ότι, ορισμένες συναρτήσεις, ανεξαρτήτως API, έχουν εσωτερικούς ελέγχους-flag π.χ η συνάρτηση alutCreateBufferFromFile χρησιμοποιεί το flag `AL_NONE`,που σημαίνει ότι αν δεν έχει φορτωθεί τίποτα στην μεταβλητή buffer τότε έχει προκύψει ένα λάθος, ανεξαρτήτως αν αυτό το λάθος είναι ανιχνεύσιμο από την alutGetError. Ο προγραμματιστής μπορεί να γνωρίζει την ύπαρξη του, που μπορεί να οφείλεται και σε κάποιο λογικό λάθος του προγράμματος. Τέτοιου τύπου flag διαθέτουν οι περισσότερες συναρτήσεις του OpenAL API και διευκολύνουν πολύ έναν προγραμματιστή που προσπαθεί να εντοπίσει ένα σφάλμα. [35,36]

Εικόνα 8 Γελοιογραφία : Επιτυχής ανίχνευση σφάλματος

3. ALC API: Audio Library Context

Σε αυτό το τμήμα της πτυχιακής, θα αναλυθούν οι βασικές λειτουργίες του ALC API. Το *ALC API* ή αλλιώς το *Audio Library Context* είναι το API που διαχειρίζεται τα *OpenAL contexts,* τα οποία είναι υπεύθυνα για τους πόρους του συστήματος, καθώς και το κλείδωμα ή ξεκλείδωμα αυτών. Επίσης το API αναλαμβάνει την διαχείριση των διεργασιών (precessing και threading). Το Context είναι ένα ελάχιστο σύνολο από δεδομένα, που περιέχει και τοποθετεί ένα υπόδειγμα βασικών δηλώσεων για την AL. [35,36,46]

Η ACL είναι επίσης υπεύθυνη και για την διαχείριση των Devices. Ένα Device μπορεί να είναι ένα hardware device, μια άλλη εφαρμογή ή μια εφαρμογή του λειτουργικού συστήματος. [35,36,]

3.1 Βασική δομή του Initializing και Exiting

Η αρχικοποίηση είναι η πρώτη διαδικασία που πρέπει να κάνει ένας προγραμματιστής, ώστε να δουλέψει με την OpenAL. Η βασική δομή του αρχικοποίησης φαίνεται παρακάτω:

1. *Σύνδεση με ένα Device*
2. *Δημιουργία ενός Context*
3. *Ενεργοποίηση του Context*

Η έξοδος είναι η τελευταία διαδικασία και περιέχει ακριβώς την αντίστροφη δομή:

1. *Εύρεση του τρέχοντος Context*
2. *Εύρεση του Device που λειτουργεί το τρέχον Context*
3. *Μετατροπή του τρέχοντος context σε NULL*
4. *Διαγραφή του Context*
5. *Αποσύνδεση από το Device*

Σε καμία περίπτωση οι δύο βασικές αυτές δομές δεν μπορούν να περιέχουν κάτι λιγότερο. Μπορούν όμως να εμπλουτιστούν ανάλογα με πρόσθετες εντολές, εφόσον είναι επιθυμητή η χρήση των APIs EFX και ALUT. [35,36]

Εικόνα 9 Binary Code

3.2 Διαχείριση των Devices

Η OpenAL παρέχει την δυνατότητα σε ένα πρόγραμμα να γνωρίζει ποιο device είναι κάθε φορά προεπιλεγμένο από τον χρήστη για την αναπαραγωγή του ήχου. Ένα πρόγραμμα μπορεί να ανοίξει ένα προεπιλεγμένο (default) device, ορίζοντας ως NULL την παράμετρο του device που θα συνδεθεί με το πρόγραμμα. Επιπλέον υπάρχει η δυνατότητα επιλογής του επιθυμητού device. [35,36]

Το άνοιγμα ή αλλιώς η σύνδεση με έναν device επιτυγχάνεται μέσω της εντολής : [35,36]

```
ALCdevice * alcOpenDevice (const ALCchar *deviceSpecifier);
```

Αντίστοιχα το κλείσιμο ή αλλιώς η αποσύνδεση με έναν device επιτυγχάνεται μέσω της εντολής : [35,36]

```
ALCboolean alcCloseDevice( ALCdevice *device);
```

Αν επιθυμούμε την σύνδεση με το default device του χρήστη, θα μπορούσαμε να γράψουμε τον παρακάτω κώδικα: [35,36]

```
int main()
{
ALCdevice *pDevice = NULL;

pDevice = alcOpenDevice(NULL);

if (pDevice != NULL) {
      printf ( "\n Open default device...\n" );
}
else {
printf ( "Δεν βρέθηκε κανένας ελεγκτής ήχου στο σύστημά σας" );
}

}
```

Αν πάλι επιθυμούμε την σύνδεση με ένα συγκεκριμένο device, θα μπορούσαμε να γράψουμε τον παρακάτω κώδικα: [35,36]

```
int main()
{
ALCdevice *pDevice = NULL;

ALCubyte DeviceName[] = "DirectSound3D";

pDevice = alcOpenDevice(DeviceName);

if (pDevice != NULL) {
      printf ( "\n Open default device...\n" );
}
else {
printf ( "Δεν βρέθηκε κανένας ελεγκτής ήχου στο σύστημά σας" );
}
}
```

Όπως βλέπουμε η OpenAL θα επικοινωνεί με την συσκευή ήχου μέσω του DirectSound3D το οποίο είναι επίσης ένα sound API. [35,36]

Πολύ σημαντικό σημείο στον κώδικα και στις δύο περιπτώσεις είναι το περιεχόμενο της εντολής if :

```
if (pDevice != NULL)
```

Αν η εντολή alcOpenDevice επιστρέψει την τιμή NULL κανένα sound driver/device δεν έχει βρεθεί στο σύστημα του χρήστη και επομένως η OpenAL δεν μπορεί να λειτουργήσει. [35,36]

Και στις δύο περιπτώσεις το κλείσιμο του Device αντιστοιχεί στον ακόλουθο κώδικα:

```
alcCloseDevice(pDevice);
```

Η επιστροφή της συνάρτησης μπορεί να είναι ή ALC_TRUE ή ALC_FALSE. Το ALC_FALSE υποδηλώνει ότι η αποσύνδεση–κλείσιμο του device δεν έγινε επιτυχώς. [35,36]

Η απαρίθμηση των devices επιτυγχάνεται με την κλήση alcGetString. Όπως έχουμε προαναφέρει, η συνάρτηση alcGetString χρησιμοποιείται και για τα ALC errors : [35,36]

```
const ALCchar * alcGetString(ALCdevice *device, ALenum param);
```

Η παράμετρος param μπορεί να αντιστοιχεί είτε στην τιμή ενός σφάλματος που θέλουμε να εκτυπώσουμε, είτε σε μια από τις ακόλουθες παραμέτρους: [35,36]

ALC string Query /alcGetString	
ALC_DEFAULT_DEVICE_SPECIFIER	Το όνομα του προσδιορισμένου default device
ALC_DEVICE_SPECIFIER	Το όνομα του προσδιορισμένου device ή ένας κατάλογος όλων των διαθέσιμων device
ALC_EXTENSIONS	Ένας κατάλογος των διαθέσιμων context επεκτάσεων (extensions).
ALC_CAPTURE_DEFAULT_DEVICE_SPECIFIER	Το όνομα του προσδιορισμένου default capture device
ALC_CAPTURE_DEVICE_SPECIFIER	Το όνομα του προσδιορισμένου capture device ή ένας κατάλογος όλων των διαθέσιμων capture device

Αν παραδείγματος χάριν θέλουμε να εκτυπώσουμε το όνομα του default device θα χρησιμοποιήσουμε την ακόλουθη γραμμή : [35,36]

```
printf ( "Default Device is %s\n",
          alcGetString(pDevice, ALC_DEFAULT_DEVICE_SPECIFIER));
```

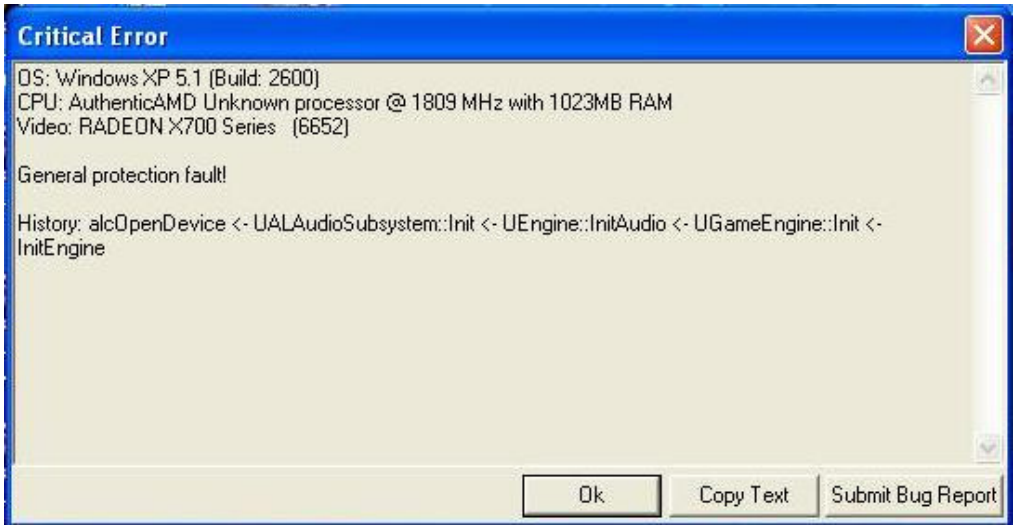
Αν τώρα θέλουμε να πάρουμε το default device από την alcGetString και με αυτό το string να ανοίξουμε το default device, θα πρέπει να γράψουμε τις ακόλουθες γραμμές κώδικα : [35,36]

```
ALCdevice *pDevice = NULL;

const ALCchar *defaultdevice = NULL;

defaultdevice = alcGetString(pDevice, ALC_DEFAULT_DEVICE_SPECIFIER);

pDevice = alcOpenDevice(defaultdevice);
```

Ο κώδικας αυτός εκτελεί ακριβώς την ίδια διαδικασία με τον παρακάτω κώδικα: [35,36]

```
ALCdevice *pDevice = NULL;

pDevice = alcOpenDevice(NULL);
```

Περισσότερες πληροφορίες για τον τρόπο με τον οποίο μπορούμε να διαχειριστούμε τα devices ενός συστήματος, καθώς και το πώς μπορούμε να απαριθμούμε τα ονόματα των διαθέσιμων devices, παρέχονται μέσω του source code στο *Κεφαλαίο 3 1. Example: Router test*



Εικόνα 10 Windows Message Box Critical Error : alcOpenDevice από το παιχνίδι America's army

3.3 Διαχείριση των Contexts

Όλες οι λειτουργίες του AL core API έχουν επιπτώσεις σε ένα τρέχον context Al. Μέσα από την ματιά της AL, το ALC είναι υπονοούμενο - δεν είναι δηλαδή ορατό ως κάποιο προγραμματιστικό αντικείμενο (π.χ. handle ή παράμετρος λειτουργίας). Μόνο ένα context AL ανά διαδικασία μπορεί να είναι ενεργό ανά κάθε στιγμή. Οι εφαρμογές που διαθέτουν πολλαπλά context Al, είτε χρησιμοποιούν threads είτε όχι, πρέπει κάθε φορά να θέτουν ένα context σε λειτουργία. [35,36]

Η εφαρμογή μπορεί να επιλέξει και να διευκρινίσει ορισμένες ιδιότητες για ένα context στο σημείο δημιουργίας του. Οι ιδιότητες που δεν διευκρινίζονται ρητά από την εφαρμογή κατά την δημιουργία, λαμβάνουν τις προεπιλεγμένες-defaults τιμές. [35,36]

ALC Context Creation Attributes	
ALC_FREQUENCY	Συχνότητα για τη μίξη του output buffer, σε Hz
ALC_REFRESH	Συχνότητα ανανέωσης σε Hz
ALC_SYNC	Flag, που υποδηλώνει ένα σύγχρονο context
ALC_MONO_SOURCES	Υποδεικνύει πόσες πηγές μπορούν να υποστηριχθούν από mono data
ALC_STEREO_SOURCES	Υποδεικνύει πόσες πηγές μπορούν να υποστηριχθούν από stereo data

Ένα context δημιουργείται χρησιμοποιώντας την alcCreateContext. Η παράμετρος device πρέπει να είναι ένα έγκυρο device. Ο κατάλογος ιδιοτήτων (attrList) μπορεί να είναι NULL ή μία λίστα ζευγών ακέραιων αριθμών, που αποτελείται από τις παραπάνω έ ιδιότητες και τις αντίστοιχες τιμές τους. [35,36]

```
ALCcontext * alcCreateContext (ALCdevice *device, const ALCint* attrlist);
```

Η δημιουργία του context θα αποτύχει στις ακόλουθες περιπτώσεις: [35,36]

- Εάν η εφαρμογή ζητά ιδιότητες που δεν είναι παρεχόμενες
- Εάν ο συνδυασμός των ιδιοτήτων δεν υποστηρίζεται
- Εάν μια ιδιότητα, ή ο συνδυασμός ιδιοτήτων δεν ταιριάζει με τις προκαθορισμένες τιμές για τον προκαθορισμό ιδιοτήτων

Εάν η δημιουργία του context αποτύχει, η συνάρτηση θα επιστρέψει NULL. Εφόσον έχει δημιουργηθεί ένα έγκυρο device και μετά την χρήση της εντολής alcCreateContext, μπορούμε να πραγματοποιήσουμε και έλεγχο για σφάλματα, όπως φαίνεται στο παρακάτω παράδειγμα: [35,36]

```
ALCcontext *pContext = NULL;

ALCdevice   *pDevice = NULL;

ALCenum      Error; /* save error status*/

pDevice = alcOpenDevice(NULL); /*Open default device*/
```

```
if (pDevice){ /*pDevice != NULL*/
      printf ( "\n Open default device...\n" );

      /*creates a context using a default device*/
      pContext = alcCreateContext(pDevice, NULL);

      if (pContext){/*pContext != NULL*/
      printf ( "\nCreates a context...\n\n" );
         }
      else {/*if pContext = NULL*/
            printf ( "\nFAILED SET ACTIVE CONTEXT!\n" );
            Error = alcGetError(pDevice);
            if (Error != ALC_NO_ERROR){
                  printf ( "Unable Active OpenAL context %s\n",
                        alGetString( Error ) );
               }
            }
      }

else {/*if pDevice = NULL*/
      printf ( "\n FAILED TO OPEN DEFAULT DEVICE!\n" );
      printf ( "No sound driver/device has been found!\n" );
     }

}
```

Η alcGetError σε περίπτωση σφάλματος κατά την εκτέλεση της εντολής alcCreateContext, μπορεί να επιστρέψει τους ακόλουθους τύπους τυποποιημένων σφαλμάτων: ALC_INVALID_VALUE και ALC_INVALID_DEVICE. [35,36]

Για να κάνουμε το context τρέχον context, δηλαδή να το ενεργοποιήσουμε, χρησιμοποιούμε την συνάρτηση alcMakeContextCurrent. Η παράμετρος context μπορεί να είναι NULL ή ένας έγκυρος δείκτης context. Η ενεργοποίηση που πραγματοποιείται ισχύει για το device που δημιουργήθηκε το context . Το NULL χρησιμοποιείται ως παράμετρος όταν δεν θέλουνε κανένα context να είναι τρέχον, κάτι το οποίο είναι χρήσιμο κατά το exiting της OpenAL και θα το δούμε παρακάτω. [35,36]

ALCboolean alcMakeContextCurrent (ALCcontext *context);

Η επιστρεφόμενη αξία μπορεί να είναι ALC_TRUE ή ALC_FALSE και απεικονίζει εάν ένα λάθος εμφανίστηκε ή όχι στην κλήση. Οι όροι τυποποιημένων σφαλμάτων κατά τη διάρκεια της εκτέλεσης αυτής της κλήσης μπορεί να είναι ALC_INVALID_CONTEXT. [35,36]

Για κάθε διεργασία λειτουργικού συστήματος μόνο ένα context μπορεί να είναι τρέχον οποιαδήποτε στιγμή. Όλες οι εντολές AL ισχύουν για το τρέχον context. Οι εντολές που έχουν επιπτώσεις σε αντικείμενα κοινά μεταξύ των contexts (π.χ. buffers) έχουν παρενέργειες σε όλα τα contexts όπως είναι λογικό. [35,36]

Το τρέχον context είναι το μόνο αποδεκτό context για την δήλωση των αλλαγών από τις εντολές AL, δηλαδή είναι το μοναδικό context που μπορεί να υποβληθεί σε

επεξεργασία (εκτός βέβαια από την επιρροή των κοινών αντικειμένων – buffers). Για να δηλωθεί ότι ένα context μπορεί να υποβληθεί σε επεξεργασία μπορεί να χρησιμοποιηθεί η συνάρτηση alcProcessContext. [35,36]

```
void alcProcessContext (ALCcontext *context);
```

Επαναλαμβανόμενες κλήσεις της alcProcessContext δεν έχουν επίπτωση σε ένα context που είναι ήδη χαρακτηρισμένο ως επεξεργάσιμο. Η προεπιλογή για ένα context που δημιουργείται από την alcCreateContext είναι ότι είναι επεξεργάσιμο. [35,36]

Αντίστοιχα μπορούμε να αναστείλουμε οποιοδήποτε context από την επεξεργασία (συμπεριλαμβανομένου και του τρέχοντος). Για να δείξουμε ότι ένα context πρέπει να ανασταλεί από την επεξεργασία, μπορούμε να χρησιμοποιήσουμε την συνάρτηση alcSuspendContext. [35,36]

```
void alcSuspendContext (ALCcontext *context);
```

Επαναλαμβανόμενες κλήσεις της alcSuspendContext δεν έχουν επίπτωση σε ένα context που είναι ήδη χαρακτηρισμένο ως ανασταλμένο. [35,36]

Το τυποποιημένο σφάλμα κατά τη διάρκεια της εκτέλεσης των δύο αυτών κλήσεων είναι το ALC_INVALID_CONTEXT. [35,36]

Τελειώνουμε το κεφαλαίο της διαχείρισης των contexts με το πως διαγράφουμε ένα context. Για να διαγράψουμε ένα context χρησιμοποιούμε την συνάρτηση alcDestroyContext. [35,36]

```
void alcDestroyContext (ALCcontext *context);
```

Ο σωστός τρόπος για να διαγραφεί ένα context δεν περιέχει μόνο την συγκεκριμένη κλήση. Οι εφαρμογές δεν πρέπει να προσπαθήσουν να διαγράψουν ένα τρέχον context. Η προσπάθεια διαγραφής ενός τρέχοντος context δεν θα λειτουργήσει και θα οδηγήσει σε ένα λάθος ALC_INVALID_OPERATION. Κατά τη διάρκεια της διαγραφής ενός context, όλες οι πηγές μέσα σε αυτό θα διαγραφούν αυτόματα. [35,36]

Οι σωστή διαδικασία διαγραφής απελευθερώνει πρώτα το τρέχον context από την χρήση, χρησιμοποιώντας την συνάρτηση alcMakeCurrent με ένα NULL context και στην συνεχεία διαγράφει το context. [35,36]

Το τυποποιημένο σφάλμα κατά τη διάρκεια της εκτέλεσης της κλήσης διαγραφής είναι το ALC_INVALID_CONTEXT. [35,36]

3.4 Συναρτήσεις ερωτήσεων

Η ALC περιέχει και συναρτήσεις που εκτελούν ερωτήσεις. Οι συναρτήσεις αυτές είναι παρά πολύ σημαντικές για την λειτουργία της OpenAL. Μια τέτοια συνάρτηση είναι και η alcGetError, που έχει ήδη αναλυθεί, όπως επίσης και η alcGetString. [35,36]

Μια εφαρμογή, χρησιμοποιώντας την alcGetCurrentContext, μπορεί να ρωτήσει και να λάβει το τρέχον context για την εφαρμογή. Εάν δεν υπάρχει κανένα τρέχον context, θα επιστραφεί το NULL. [35,36]

```
ALCcontext * alcGetCurrentContext (ALCvoid);
```

Επίσης μπορεί να κάνει ερώτηση και να λάβει το device ενός δεδομένου context. [35,36]

```
ALCdevice* alcGetContextsDevice (ALCcontext *context);
```

Το τυποποιημένο σφάλμα κατά τη διάρκεια αυτής της κλήσης είναι το ALC_INVALID_CONTEXT. [35,36]

Για να ελέγξει ότι μια δεδομένη επέκταση[1] είναι διαθέσιμη για το τρέχον context και device χρησιμοποιείται η κλίση. [35,36]

```
ALCboolean alcIsExtensionPresent (ALCdevice *device, const ALCchar *extname);
```

Αν δοθεί μια άκυρη η μη υποστηρίξιμη επέκταση θα επιστρέψει ALC_FALSE, ενώ μπορεί να δεχτεί την τιμή NULL ως device. Ένα NULL extname θα οδηγήσει σε ένα λάθος ALC_INVALID_VALUE και η επιστρεφόμενη τιμή θα είναι ALC_FALSE. Το Extname δεν χρειάζεται να είναι γραμμένο σε κεφαλαία ή μικρά, η συνάρτηση θα μετατρέψει το Extname σε κεφαλαία εσωτερικά. [35,36]

Επίσης μπορεί να ερωτήσει για έλεγχο της δυνατότητας εύρεσης της διεύθυνσης μνήμης μιας context extension function ή μιας λειτουργίας πυρήνα. Αυτά μπορούν να ανακτηθούν χρησιμοποιώντας alcGetProcAddress. [35,36]

```
void * alcGetProcAddress( ALCdevice *device, const ALCchar *funcname );
```

Η χρησιμοποίηση ενός NULL device δεν εγγυάται ότι θα επιστραφεί η διεύθυνση, ενώ το context δεν παίζει κανένα απολύτως ρόλο στην συγκεκριμένη κλήση. Επίσης η χρήση του NULL ως παραμέτρου ονόματος (funcname) θα προκαλέσει ένα λάθος ALC_INVALID_VALUE. [35,36]

[1] Παραδείγματα υλοποίησης υπάρχουν στο υποκεφάλαιο 6. Example: System info - Extensions

3.6 Υλοποίηση - Initializing και Exiting

Στο υποκεφάλαιο 3.1 δόθηκε η θεωρητική δομή του initializing και του exiting. Στο τελευταίο αυτό υποκεφάλαιο και έχοντας αναλύσει τις περισσότερες συναρτήσεις της ALC, μπορεί να γίνει επέκταση και στον κώδικα του initializing και του exiting, συνοψίζοντας έτσι τις σημαντικότερες λειτουργίες του API. [35,36]

Initializing

Η πιο απλή μορφή ενός ολοκληρωμένου κώδικα initializing μιας εφαρμογής φαίνεται παρακάτω:

```
ALCcontext *pContext = NULL;
ALCdevice  *pDevice = NULL;

/*Σύνδεση με το default device*/
pDevice = alcOpenDevice(NULL);

/*Δημιουργία ενός context χρησιμοποιώντας το default device*/
pContext = alcCreateContext(pDevice, NULL);

/*Ενεργοποίηση του context για επεξεργασία – θέτει το context ως το
τρέχον context*/
alcMakeContextCurrent(pContext);
```

Αυτό το μικρό τμήμα κώδικα κάνει την OpenAL initialize και είναι απαραίτητο για να λειτουργήσει η OpenAL. Εφόσον έχουν εκτελεστεί σωστά οι παραπάνω εντολές, το πρόγραμμα είναι έτοιμο να χρησιμοποιήσει το AL API (για την χρήση της ALUT και EFX, όπως έχει προαναφερθεί, χρειάζεται περαιτέρω επεξεργασία, που θα αναλυθεί ξεχωριστά στο υποκεφάλαιο που θα αναφέρεται σε κάθε API). [35,36]

Exiting

Η πιο απλή μορφή ενός ολοκληρωμένου κώδικα exiting μιας εφαρμογής φαίνεται παρακάτω:

```
ALCcontext *pContext;
ALCdevice   *pDevice;

/*Εύρεση του τρέχοντος context*/
pContext = alcGetCurrentContext();

/*Εύρεση του device που λειτουργεί το τρέχον context*/
pDevice = alcGetContextsDevice(pContext);

/*Μετατροπή του τρέχοντος context σε NULL*/
alcMakeContextCurrent(NULL);

/*Διαγραφή του context*/
alcDestroyContext(pContext);

/*Αποσύνδεση από το device*/
alcCloseDevice(pDevice);
```

Τα δύο παραδείγματα είναι πλήρως υλοποιήσιμα και σε ένα σύστημα χωρίς error δεν θα δημιουργήσουν κανένα πρόβλημα. Δεν χρησιμοποιήθηκε κανένα flag και εξέταση για errors, μιας και τέτοια παραδείγματα υπάρχουν σε ολοκληρωμένο κώδικα στο ΚΕΦΑΛΑΙΟ 3.

Επίσης πλήρης κατάλογος με όλες των λειτουργιών της ALC παρουσιάζεται στο Παράρτημα 1:

ΠΙΝΑΚΑΣ 1.1 ALC Functions
ΠΙΝΑΚΑΣ 1.2 ALC Primitive Types
ΠΙΝΑΚΑΣ 1.3 ALC Define Values

4. AL API: Audio Library - Core OpenAL

Το AL API διαθέτει τρεις βασικές κατηγορίες αντικειμένων: [35,36]

- Ένα μοναδικό *Listener* ανά context.
- Πολλαπλά *Sources* σε κάθε context.
- Πολλαπλούς *buffers* που είναι κοινοί για όλα τα context αλλά για ένα device.

Τα αντικειμένα AL, στην μεγάλη τους πλειοψηφία, είναι δυναμικά και δημιουργούνται μετά από αίτηση του προγράμματος (π.χ. Το πρόγραμμα ζητά την δημιουργία ενός buffer).Υπάρχουν επίσης αντικείμενα AL που δεν είναι απαραίτητο να δημιουργηθούν, και δεν μπορούν να δημιουργηθούν, μετά από την αίτηση του προγράμματος. Αυτήν την περίοδο, ο Listener είναι το μοναδικό *στατικό* αντικείμενο της OpenAL. [35,36]

Τα δυναμικά αντικείμενα καλούνται χρησιμοποιώντας έναν ακέραιο αριθμό τύπου ALuint, ο οποίος αναφέρεται ως *«όνομα»* του αντικειμένου. Τα ονόματα μπορούν να ισχύσουν ανεξαρτήτως του context που δημιουργήθηκαν, εάν τα εν λόγω αντικείμενα μπορούν να μοιραστούν μεταξύ των contexts. Δεν υπάρχει κανένας περιορισμούς για τις ακριβείς τιμές (ονόματα) των αντικειμένων. Η τιμή (όνομα) πρέπει να είναι διαφορετική για κάθε αντικείμενο ίδιας κατηγορίας, αλλά δεν υπάρχει κανένας περιορισμός για διαφορετικές κατηγορίες αντικειμένων. Τα αντικείμενα που είναι μοναδικά, όπως ο Listener, δεν απαιτούν και δεν έχουν ένα "όνομα".[35,36]

Η AL παρέχει κλήσεις δημιουργίας και μέσω των κλήσεων αυτών λαμβάνει τα ονόματα των αντικειμένων. Ένα πρόγραμμα μπορεί να ζητήσει την δημιουργία πολλαπλών αντικειμένων μιας δεδομένης κατηγορίας, χρησιμοποιώντας alGen{Object}s. [35,36]

```
void alGen{Object}s (ALsizei n, ALuint *Names);
```

όπου *n* είναι ο αριθμός των αντικειμένων που θα δημιουργηθούν.

Έτσι για την δημιουργία αντικειμένων buffers και Sources μπορούμε να χρησιμοποιήσουμε της αντίστοιχες συναρτήσεις. [35,36]

```
void alGenBuffers (ALsizei n, ALuint *bufferNames);

void alGenSources (ALsizei n, ALuint *sourceNames);
```

Η αίτηση για δημιουργία μηδενικού αριθμού ονομάτων δεν θα λειτουργήσει. Η AL θα αποκριθεί με ένα λάθος AL_INVALID_VALUE εάν δεν μπορεί να αποθηκεύσει τα *n* ονόματα στη δεδομένη σειρά ή εάν δεν μπορεί να παραγάγει το ζητούμενο αριθμό αντικειμένων, λόγω περιορισμού των πόρων μνήμης. Ένα λάθος AL_OUT_OF_MEMORY θα παρουσιαστεί εάν δεν μπορεί να διαθέσει τα αντικείμενα, λόγω έλλειψης της διαθέσιμης μνήμης. [35,36]

Η AL «απελευθερώνει» τα αντικείμενα χρησιμοποιώντας την alDelete{Object}s, ζητώντας τη διαγραφή των αντικειμένων που συνδέονται με τα αντίστοιχα ονόματα. [35,36]

```
void alDeleteBuffers (ALsizei n, ALuint *bufferNames);

void alDeleteSources (ALsizei n, ALuint *sourceNames);
```

Ένα source αντικείμενο που εκτελεί ένα ήχο μπορεί να διαγραφεί. Ουσιαστικά θα σταματήσει αυτόματα και θα διαγραφεί έπειτα, ενώ ένας buffer που είναι συνδεμένος με μια πηγή δεν μπορεί να διαγραφεί. [35,36]

Μόλις διαγραφούν τα αντικείμενα, δεν ισχύουν πλέον για καμία απολύτως λειτουργία της OpenAL. Εάν ένα ή περισσότερα από τα προσδιορισμένα ονόματα είναι μη ισχύοντα, ένα λάθος AL_INVALID_NAME θα καταγραφεί και κανένα αντικείμενο δεν θα διαγραφεί. [35,36]

Ένα πρόγραμμα μπορεί να ελέγξει εάν ένα όνομα αντικειμένου ισχύει χρησιμοποιώντας τη ερώτηση alIs{Object}. Η συνάρτηση αυτή επιστρέφει AL_TRUE, εάν το όνομα είναι ένα έγκυρο όνομα αντικειμένου και AL_FALSE σε οποιαδήποτε άλλη περίπτωση. Δηλαδή δεν μπορεί να κάνει διάκριση μεταξύ των άκυρων και διαγραμμένων ονομάτων. [35,36]

```
ALboolean alIsBuffer (ALuint bufferName);

ALboolean alIsSource (ALuint sourceName);
```

Εικόνα 11 Creative Digital Life

4.1 Εισαγωγή στην δομή των ιδιοτήτων αντικειμένων-Attributes

Για τον έλεγχο των χαρακτηριστικών των αντικειμένων, παρέχονται διάφορες εντολές, που η λειτουργία τους εξαρτάται από τις πραγματικές ιδιότητες μιας δεδομένης κατηγορίας αντικειμένου. Μια εντολή OpenAL, που μπορεί να επηρεάσει την κατάσταση ενός αντικειμένου, έχει συνήθως τη μορφή: [35,36]

```
void al{Object}{n}{if}{v}(ALuint objectName, ALenum paramName, T values );
```

Όπου :

- *{Object}* είναι το όνομα ενός αντικειμένου OpenAL(Buffer, Listener, Source).
- *{n}* Το μέγεθος των τιμών που μπορούμε να εισάγουμε στην εντολή. Για μέγεθος ένα, το n παραλείπεται.
- {if} Δείχνει τον τύπο της τιμής που εισάγουμε στην εντολή , "i" για integer, "f" για float κλπ.
- {v} Δείχνει ότι δέχεται ένα διάνυσμα του τύπου{if}.
- Ενώ το T υποδηλώνεται από τις παραμέτρους {if} και {v}.

Η παράμετρος objectName διευκρινίζει το αντικείμενο OpenAL που επηρεάζεται από αυτήν την κλήση. Η χρήση ενός άκυρου ονόματος θα προκαλέσει ένα λάθος AL_INVALID_NAME. [35,36]

Το χαρακτηριστικό του αντικειμένου που επηρεάζεται ονομάζεται paramName. Όπως έχει προαναφερθεί τα χαρακτηριστικά ενός αντικειμένου δεν είναι απαραίτητα ίδια για όλες τις κατηγορίες αντικείμενων της OpenAL. Επομένως, αν χρησιμοποιηθεί ένα χαρακτηριστικό που δεν είναι έγκυρο για το συγκεκριμένο αντικείμενο, θα προκληθεί ένα λάθος AL_INVALID_OPERATION. [35,36]

Είναι προφανές ότι δεν μπορεί να είναι έγκυρες όλες οι πιθανές τιμές(values) για ένα δεδομένο objectName και paramName. Επομένως η χρήση μιας μη έγκυρης τιμής ή ενός NULL δείκτη θα προκαλέσει ένα λάθος AL_INVALID_VALUE. [35,36]

Στην περίπτωση των μοναδικών στατικών αντικειμένων όπως ο Listener, το objectName παραλείπεται και επόμενος υπονοείται από το {Object} του ονόματος της εντολής: [35,36]

```
void al{Object}{n}{if}{v} (ALenum paramName, T values);
```

Σε αυτό το σημείο μπορούμε να δούμε την σύνταξη ορισμένων εντολών που χρησιμοποιούν τους παραπάνω κανόνες: [35,36]

- Θέτει μια floating point παράμετρο ενός buffer.
  ```
  void alBufferf(ALuint buffer, ALenum param, ALfloat value);
  ```
- Θέτει έναν διάνυσμα floating point ως παράμετρο ενός buffer.
  (π.χ. ALfloat values [] = { 0.0, 0.0, 0.0 };)
  ```
  void alBufferfv(ALuint buffer, ALenum param, ALfloat *values);
  ```
- Θέτει τρεις float παραμέτρους μιας πηγής (v1,v2,v3)

```
void alSource3f(ALuint source, ALenum param, ALfloat v1,ALfloat v2,ALfloat v3);
```

- Θέτει μια floating point παράμετρο για τον ακροατή.
  ```
  void alListenerf(ALenum param, ALfloat value);
  ```

Η AL παρέχει και εντολές ερώτησης-ανάκτησης της τρέχουσας ιδιότητας ενός δυναμικού η στατικού αντικειμένου. Οι συναρτήσεις αυτές εξαρτώνται από τις πραγματικές ιδιότητες μιας δεδομένης κατηγορίας αντικειμένου. Οι έγκυρες τιμές για την παράμετρο paramName είναι ίδιες και αντίστοιχες με εκείνες της εκχώρησης–τροποποίησης χαρακτηριστικών. Μια εντολή OpenAL, που μπορεί να ανακτήσει την κατάσταση ενός αντικειμένου, έχει συνήθως τη μορφή: [35,36]

```
void alGet{Object}{n}{if}{v} (uint objectName, enum paramName, T * destination);
```

Και σε αυτήν την περίπτωση στα μοναδικά στατικά αντικείμενα, όπως ο Listener, παραλείπουμε το objectName:

```
void alGet{Object}{n}{if}{v} (enum paramName, T * destination);
```

Η διάφορα των δύο τύπων συναρτήσεων είναι ότι στην μια θέτουμε τιμές *al{Object},* ενώ στην άλλη ανακτάμε τιμές *alGet{Object}*. [35,36]

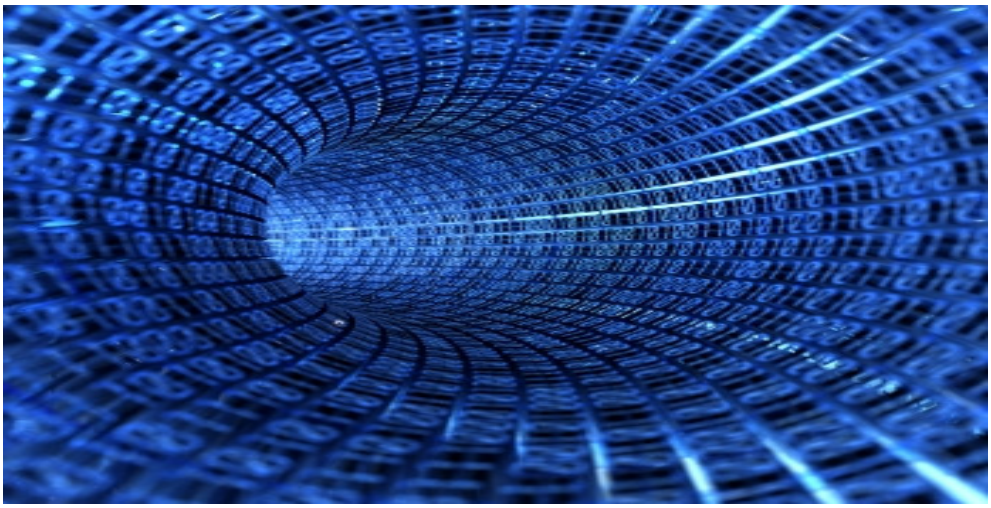
Παρακάτω φαίνεται η σύνταξη ορισμένων εντολών ερώτησης: [35,36]

- Ανακτά τρεις integer τιμές, που αντιπροσωπεύουν μια παράμετρο για τον ακροατή (v1,v2,v3).
  ```
  void alGetListener3i(ALenum param, ALint *v1,ALint *v2,ALint *v3);
  ```

- Ανακτά έναν διάνυσμα floating point, ως παράμετρο μιας πηγής.
  ```
  void alGetSourcefv(ALuint source, ALenum param, ALfloat *values);
  ```

- Ανακτά έναν διάνυσμα integer, που είναι παράμετρος ενός buffer
  ```
  void alGetBufferiv(ALuint buffer, ALenum pname, ALint *values);
  ```

Ένας πλήρης κατάλογος τέτοιων εντολών υπάρχει στο Παράρτημα1
ΠΙΝΑΚΑΣ 2.1: AL Functions



Εικόνα 12 Digital Life - Binary Code

4.3 Buffers

Οι buffers σαν αντικείμενα της OpenAL είναι υπεύθυνα για την αποθήκευση, συμπίεση και αποσυμπίεση δεδομένων. Ένα πρόγραμμα μπορεί να ζητήσει την δημιουργία αντικειμένων buffers, ώστε να τα γεμίσει με δεδομένα. Τα δεδομένα μπορούν να παρασχεθούν συμπιεσμένα και να κωδικοποιηθούν, εφ' όσον υποστηρίζεται το format. Οι Buffers μπορούν, εσωτερικά, να περιέχουν τα δεδομένα κυματομορφών, ως ασυμπίεστα ή συμπιεσμένα δείγματα. [35,36]

Όπως έχει προαναφερθεί, αντίθετα από τα αντικείμενα Sources και Listener, τα αντικείμενα buffers μπορούν να ισχύουν για πολλαπλά Context. Ένας buffer μπορεί να παραπέμπει δεδομένα σε πολλαπλά Sources. Αυτές οι δύο βασικές ιδιότητες του αντικειμένου επιτρέπουν σε ένα σύστημα την ελάχιστη δυνατή επεξεργαστική ισχύ με αποτέλεσμα την καλύτερη (πιο ελαφριά) λειτουργία. [35,36]

Ένας buffer μπορεί να βρίσκεται σε μια από της ακόλουθες καταστάσεις, σε σχέση με ένα αντικείμενο source: εκτός χρήσης (Unused), σε κατάσταση επεξεργασίας (Processed), εν αναμονή (Pending). Αυτό που έχει ιδιαίτερη σημασία είναι ότι μπορεί να διαγραφεί μόνο όταν είναι σε κατάσταση Unused. Ειδικότερα, ένας buffer που είναι συνδεδεμένος, εν αναμονή ή τρέχων, σε ένα αντικείμενο source, δεν μπορεί να διαγραφεί. [35,36]

Η κατάσταση ενός buffer εξαρτάται από την κατάσταση όλων των sources και την σειρά αναμονής που διαθέτουν. Ακόμα και όταν οι sources βρίσκονται σε κατάσταση AL_STOPPED ή AL_INITIAL, υπάρχουν οι καταχωρήσεις των σειρών αναμονής και αναγκάζουν τους buffers να βρίσκονται σε κατάσταση Processed. [35,36]

Στην επόμενη έκδοση της OpenAL αναμένεται να προστεθούν στις ιδιότητες ενός buffer, οι τρεις αυτές καταστάσεις: AL_UNUSED, AL_PENDING, AL_PROCESSED. Προς το παρόν η εύρεση της κατάστασης ενός buffer είναι δυνατή μόνο με ερώτηση μέσω του source και αναφέρεται στις ιδιότητες των sources. [35,36]

Οι ιδιότητες για τα δεδομένα που περιλαμβάνονται στον buffer είναι τέσσερεις: AL_FREQUENCY, AL_BITS, AL_CHANNELS, AL_SIZE. [35,36]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT	ΠΕΡΙΓΡΑΦΗ
AL_FREQUENCY	0 έως οποιαδήποτε	0	Συχνότητα, δειγμάτων ανά δευτερόλεπτο σε Hertz [Hz], δηλ. Συχνότητα δειγματοληψίας
AL_BITS	8, 16	16	Ο αριθμός Bits ανά δείγμα
AL_CHANNELS	1, 2	1	Ο αριθμός καναλιών.
AL_SIZE	0 έως MAX_UINT	0	Μέγεθος των bytes για τα buffer data.

Χρησιμοποιώντας τις παρακάτω εντολές μπορούμε να τροποποιήσουμε αυτές τις παράμετρους: [35,36]

```
void alBuffer{n}{if}{v} (ALuint bufferName, ALenum paramName, T *values);
```

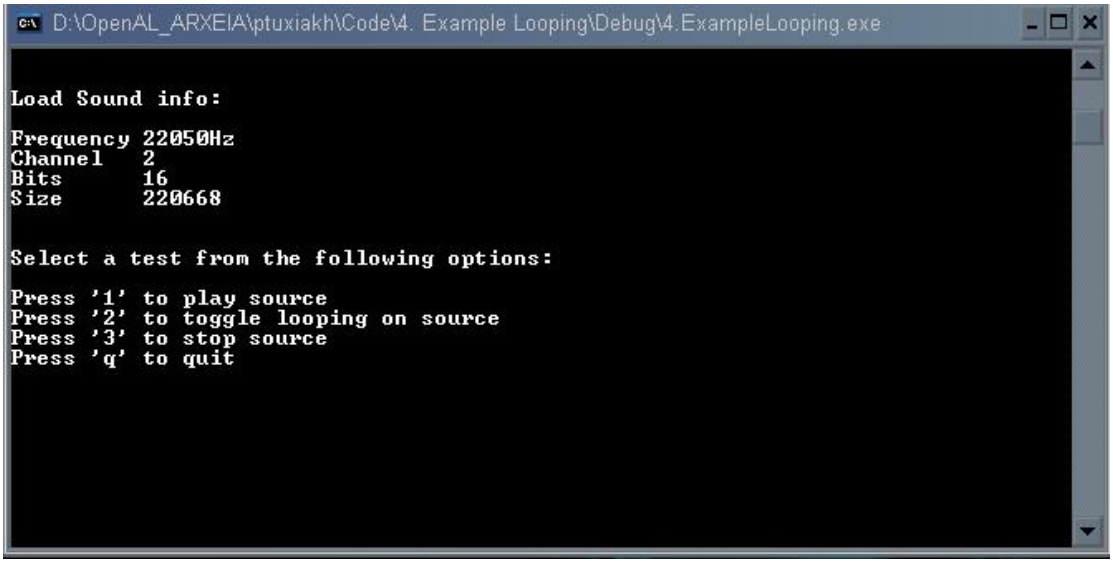
Ενώ αντίστοιχα μπορούμε να αντλήσουμε τις πληροφορίες αυτές, χρησιμοποιώντας τις αντίστοιχες εντολές : [35,36]

```
void alGetBuffer{n}{if}{v} (ALuint bufferName, ALenum paramName, T *values);
```

Παραδείγματος χάρη, από τον παρακάτω κώδικα μπορούμε να αντλήσουμε και να εκτυπώσουμε της πληροφορίες του αρχείου ήχου που είναι φορτωμένο στον buffer: [35,36]

```
ALuint    Buffer;  /*Buffer Name */
ALint     channels;/*Sound Channel info for buffer*/
ALsizei   freq;/*Sound Frequency info for buffer*/
ALint     bits;/*Sound Bits info for buffer*/
ALsizei   size;/*Sound size info for buffer*/

 /*. . . . . */

alGetBufferi(Buffer, AL_FREQUENCY, &freq);
alGetBufferi(Buffer, AL_BITS, &bits);
alGetBufferi(Buffer, AL_CHANNELS, &channels
alGetBufferi(Buffer, AL_SIZE, &size);

printf("\nFrequency%iHz", freq);
printf("\nChannel%i", channels);
printf("\nBits%i", bits);
printf("\nSize%i", size);

/*. . . . . */
```

Στο παράδειγμα 4. Example: Looping , του Κεφαλαίου 3 χρησιμοποιείται ο παραπάνω κώδικας, με αποτέλεσμα ο χρήστης να γνωρίζει αυτές τις πληροφορίες για το αρχείο ήχου που μόλις φόρτωσε και ετοιμάζεται να εκτελέσει.



Η OpenAL για τα αντικείμενα buffers διαθέτει και τέσσερα standard format: [35,36]

AL_FORMAT_MONO8	Format ήχου – mono 8 bit
AL_FORMAT_MONO16	Format ήχου – mono 16 bit
AL_FORMAT_STEREO8	Format ήχου – stereo 8 bit
AL_FORMAT_STEREO16	Format ήχου – stereo 16 bit

Η εντολή που γεμίζει έναν buffer AL με audio data, αλλά μπορεί να χρησιμοποιηθεί κατ' επέκταση για να φορτώσει και άλλους τύπους data, φαίνεται παρακάτω: [35,36]

void alBufferData(ALuint bufferName, ALenum format, const ALvoid *data, ALsizei size, ALsizei freq);

Στις παλιότερες εκδόσεις της OpenAL η εντολή alBufferData ήταν η κύρια εντολή για αυτήν την διαδικασία και χρησιμοποιόνταν μαζί με την εντολή alutLoadWAVFile του AULT API. Όμως, επειδή οι εντολές alutLoadWAV στην έκδοση 1.1 τέθηκαν σε απόσυρση (παρέχονται μόνο για την προς τα πίσω συμβατότητα), η εντολή alBufferData δεν αποτελεί πλέον τον βασικό τρόπο εκχώρησης δεδομένων. [35,36]

Ένα παράδειγμα της χρήσης της, όπως λειτουργούσε παλιά, φαίνεται παρακάτω[1]:

```
#define NUM_BUFFERS 1

ALenum format;
ALsizei size;
ALvoid* data;
ALsizei freq;
ALboolean loop;
ALuint Buffers[NUM_BUFFERS]; // Buffers hold sound data.

alGenBuffers(NUM_BUFFERS, Buffers);
// Load wav data into buffers.
alutLoadWAVFile("wavdata/thunder.wav", &format, &data, &size, &freq, &loop);
alutUnloadWAV(format, data, size, freq);
alBufferData(Buffers[THUNDER], format, data, size, freq);
```

Αντίστοιχα για την ίδια ακριβώς διαδικασία από την έκδοση 1.1 της OpenAL χρησιμοποιούμε τον παρακάτω κώδικα : [35,36]

```
ALuint              Buffer;/*Buffer Name */
Buffer = alutCreateBufferFromFile (wavdata/thunder.wav);
```

Όπως φαίνεται η πλειοψηφία των εντολών και οι παράμετροι της AL για τα αντικείμενα των buffers πλέον δεν είναι αναγκαία να χρησιμοποιηθούν. Περισσότερες πληροφορίες για την λειτουργία του load data δίνονται στο Υποκεφάλαιο 6. ALUT API: The OpenAL Utility Toolkit

[1] Η πλειοψηφία των tutorial την OpenaAL που παρέχονται στο internet, λόγω παλαιότητας, χρησιμοποιούν την παρακάτω συνάρτηση. Ο συγκεκριμένος κώδικας προήλθε από την διεύθυνση http://www.devmaster.net/articles/openal-tutorials/lesson5.php (Πρόσβαση : 28 Νοέμβριου 2007)

4.4 Sources

Τα *Sources,* ως αντικείμενα της OpenAL, διευκρινίζουν χαρακτηριστικά όπως τη θέση, την ένταση (gain) και την ταχύτητα (velocity). Υποβάλλονται σε επεξεργασία ανεξάρτητα ο ένας από τον άλλο και όταν ένας *Buffer* συνδέεται σε ένα *Source* γίνεται εφικτή η τροποποίηση, καθώς και ο έλεγχος των παραμέτρων των samples data που παρέχονται από τον *Buffer* στον *Source*. [35,36]

Οι *Sources* καθορίζουν έναν ήχο τοπικά και ορίζουν ένα σύνολο ιδιοτήτων, που εφαρμόζονται σε έναν ήχο στα πρώτα -πρώτα στάδια της επεξεργασίας του. Όλα τα χαρακτηριστικά που επιθυμεί μια εφαρμογή να προσδιορίσει –τροποποιήσει πρέπει να προσδιορίζονται προτού ο *source* τοποθετηθεί στον *listener* για την τελική επεξεργασία, μίξη και εκτέλεση. [35,36]

Η OpenAL παρέχει επίσης πρόσθετες λειτουργίες χειρισμού και ερώτησης, για την κατάσταση εκτέλεσης του *Source*, όπως το τρέχον playing status του *Source* (started, stopped και paused), ενώ σε αυτές τις λειτουργίες συμπεριλαμβάνεται και η πρόσβαση στην τρέχουσα θέση δείγματος μέσα στο συνδεδεμένο *Buffer*. [35,36]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_SOURCE_STATE	AL_INITIAL AL_PLAYING AL_PAUSED AL_STOPPED	AL_INITIAL

Μια εφαρμογή μπορεί να ρωτήσει την τρέχουσα κατάσταση οποιουδήποτε *Source,* χρησιμοποιώντας την εντολή alGetSource με το όνομα του Source και θέτοντας ως παράμετρο την AL_SOURCE_STATE. [35,36]

```
alGetSourcei(sources, AL_SOURCE_STATE, &sourceState);
```

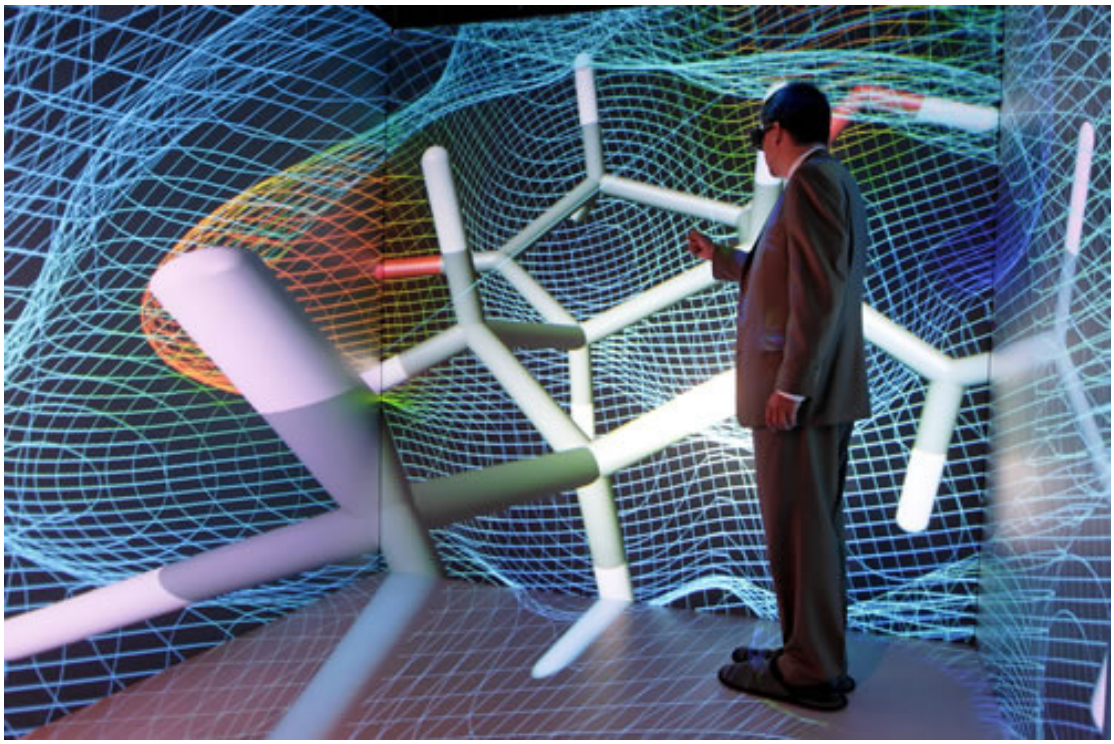
Κάθε Source μπορεί να βρίσκεται σε μία από τέσσερις πιθανές καταστάσεις εκτέλεσης: AL_INITIAL, AL_PLAYING, AL_PAUSED, AL_STOPPED. [35,36]

Οι Sources που είναι είτε AL_PLAYING, είτε AL_PAUSED θεωρούνται ενεργοί. Οι Sources που είναι είτε AL_STOPPED, είτε AL_INITIAL θεωρούνται ανενεργοί. Μόνο οι Sources που βρίσκονται σε κατάσταση AL_PLAYING συμπεριλαμβάνονται σε επεξεργασία. [35,36]

Εκτός από την ιδιότητα ερώτησης η OpenAL διαθέτει και μια σειρά από εντολές μετάβασης :

Εντολές μετάβασης κατάστασης	
alSourcePlay (ALuint sourceName);	Θέτει σε αναπαραγωγή μια πηγή - AL_PLAYING
alSourcePause(ALuint sourceName);	Θέτει σε παύση μια πηγή- AL_PAUSED
alSourceStop(ALuint sourceName);	Σταματά μια πηγή - AL_STOPPED
alSourceRewind(ALuint sourceName);	Σταματά μια πηγή - AL_INITIAL

Τρεις είναι οι βασικές ιδιότητες του αντικειμένου *Sources:* AL_POSITION, AL_GAIN και AL_VELOCITY. Αυτές οι τρεις ιδιότητες χρησιμοποιούνται ακριβώς με τον ίδιο τρόπο και για το αντικείμενο *Listener*. Η διαφορά είναι ότι στην μια περίπτωση επηρεάζουν το αντικείμενο *Source,* ενώ στην άλλη το αντικείμενο *Listener.* Αυτές οι τρεις βασικές ιδιότητες αποτελούν την βάση για ένα τρισδιάστατο ακουστικά εικονικό χώρο. [35,36]



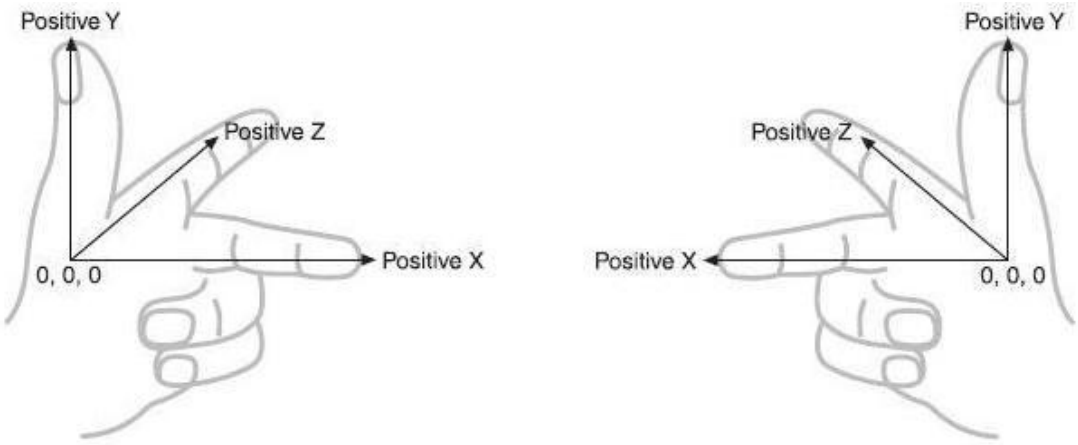
Εικόνα 13 CAVE Virtual Reality: Οι τεχνικοί της Sun Microsystems και Fakespace Systems δημιούργησαν αυτό το εικονικό περιβάλλον που παρομοιάζεται με Holodeck στο Star Trek!

Το AL_POSITION διευκρινίζει την τρέχουσα θέση του αντικειμένου στον τρισδιάστατο χώρο. Μπορεί να δεχτεί οποιαδήποτε 3-tuple float τιμή, εκτός από NaN και άπειρο. Μπορεί να περιγραφεί μέσα από ένα floating-point διάνυσμα "fv"(alGetSourcefv ή alSourcefv) ή από τρεις μεμονωμένες floating-point τιμές "3f"(alGetSource3f ή alSource3f). [35,36]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT	ΠΕΡΙΓΡΑΦΗ
AL_POSITION	Οποιαδήποτε αριθμητική τιμή εκτός από NaN[1]	{0.0f, 0.0f, 0.0f}	Η τρέχουσα θέση σε τρισδιάστατο χώρο.

[1] NaN (**N**ot **a** **N**umber), περισσότερες πληροφορίες παρέχονται στην διεύθυνση http://en.wikipedia.org/wiki/NaN

Η OpenAL (σαν την OpenGL) χρησιμοποιεί ένα δεξιόχειρο καρτεσιανό ισοδύναμο σύστημα RHS (Right-Handed System) για την περιγραφή του τρισδιάστατου χώρου. Η μετατροπή του σε ένα αριστερόχειρο ισοδύναμο σύστημα (LHS) επιτυγχάνεται μέσω της αλλαγής κατεύθυνσης του Z. [35,47]



Εικόνα 14 Αριστερά παρουσιάζεται ένα Left-handed coordinate system(LHS), ενώ δεξιά ένα Right-handed coordinate system(RHS)

Αυτό που έχει ιδιαίτερη σημασία να γνωρίζουμε, είναι ότι το κάθε διάνυσμα αντιπροσωπεύει ένα σημείο και ότι το κέντρο είναι το σημείο {0.0f, 0.0f, 0.0f}. Με αυτό τον τρόπο, τοποθετούμε ηχητικά αντικείμενα σε διάφορα σημεία ενός εικονικού κόσμου. Αν θέλαμε ένα αντικείμενο να το τοποθετήσουμε αριστερά –πάνω, θα δίναμε στην OpenAL τις ακόλουθες τιμές:{-1.0f, 1.0f, 0.0f}. Ο τρόπος που θέτουμε τα θετικά και τα αρνητικά ακολουθούν τα πρότυπα της GL, δηλαδή «πάνω Y», «μέσα Z» και «δεξιά X» είναι θετικές τιμές. [35,47]

Το AL_VELOCITY διευκρινίζει το τρέχον velocity (ταχύτητα και κατεύθυνση) του αντικειμένου στο καρτεσιανό ισοδύναμο σύστημα. Είναι μια διανυσματική φυσική ποσότητα, που μπορεί να δεχτεί οποιαδήποτε 3-tuple float/double τιμή εκτός από NaN. [35,48]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT	ΠΕΡΙΓΡΑΦΗ
AL_VELOCITY	Οποιαδήποτε αριθμητική τιμή εκτός από NaN	{0.0f, 0.0f, 0.0f}	Καθορίζει το τρέχον velocity σε τρισδιάστατο χώρο

Η ιδιότητα AL_VELOCITY δεν έχει επιπτώσεις στη θέση του *Source*. Η AL_VELOCITY δεν υπολογίζει την ταχύτητα του ήχου από τις επόμενες θέσεις, ούτε ρυθμίζει τη θέση κατά τη διάρκεια του χρόνου, βασισμένη στη διευκρινισμένη ταχύτητα. Το διάνυσμα αυτό παρέχεται ώστε να μπορεί να επιτευχθεί οποιοσδήποτε υπολογισμός και ρύθμιση από τον υλοποιητή μιας εφαρμογής. [35,48]

Επομένως αν θέλαμε να αρχικοποιήσουμε τις default τιμές για τις βασικές ιδιότητες ενός *Source,* θα χρησιμοποιούσαμε τον ακόλουθο κώδικα: [35,36]

```
ALuint Source;

// Position of the source sound.
ALfloat SourcePos[] = { 0.0, 0.0, 0.0 };
// Velocity of the source sound.
ALfloat SourceVel[] = { 0.0, 0.0, 0.0 };

alSourcef (Source, AL_GAIN,     1.0      );
alSourcefv(Source, AL_POSITION, SourcePos);
alSourcefv(Source, AL_VELOCITY, SourceVel);
```

Εικόνα 15 Τα συγκεκριμένα 3d που κατασκευάστηκαν στο περιβάλλον blender ως ηχητικά αντικείμενα, θα μπορούσαν να αποτελούν ένα Source.

Το AL_GAIN, όπως είναι προφανές, καθορίζει την ένταση (Gain). Σαν ιδιότητα source ισχύει για εκείνο το συγκεκριμένο source που εφαρμόσθηκε. Σαν ιδιότητα listener ισχύει αποτελεσματικά για όλα τα sources στο τρέχον context. Η προεπιλογή 1.0 σημαίνει ότι ο ήχος δεν υφίσταται καμία τροποποίηση. Μια τιμή AL_GAIN 0.5 ισοδυναμεί με μείωση κατά 6 DB, ενώ τιμή μηδέν είναι ίση με 0DB. [35]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT	ΠΕΡΙΓΡΑΦΗ
AL_GAIN	[0, Οποιαδήποτε]	1.0f	Καθορίζει μια κλιμακωτή διαμόρφωση πλάτους. (Ένταση)

Επίσης η OpenAL παρέχει παραμέτρους για το Min και το Max Gain. Το AL_MIN_GAIN δείχνει το ελάχιστο AL_GAIN, ενώ το AL_MAX_GAIN δείχνει το μέγιστο AL_GAIN. [35]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_MIN_GAIN	[0.0f, 1.0f]	0.0f

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_MAX_GAIN	[0.0f, 1.0f]	1.0f

Οι τιμές αυτές μπορούν να χρησιμοποιηθούν στο τέλος της επεξεργασίας των διαφόρων παραγόντων μείωσης που βασίζονται στην απόσταση και του Source AL_GAIN. Το GAIN που υπολογίζεται συγκρίνεται με αυτές τις τιμές. Εάν το GAIN είναι χαμηλότερο από AL_MIN_GAIN θα τοποθετηθεί η τιμή AL_MIN_GAIN. Αντίστοιχα αν το GAIN είναι υψηλότερο από AL_MAX_GAIN θα τοποθετηθεί η τιμή AL_MAX_GAIN. Αυτή η διαδικασία υπολογίζεται προτού να εφαρμοστεί το GAIN του listener. [35]

Εάν τοποθετηθεί ένα μηδενικό AL_MIN_GAIN, το GAIN δεν θα διορθωθεί σύμφωνα με αυτήν την τιμή. Αντίθετα εάν τοποθετηθεί ένα μηδενικό AL_MAX_GAIN, οι Sources θα μειωθούν στα 0DB. [35]

Όπως έχει προαναφερθεί, χρησιμοποιώντας τον παρακάτω τύπο εντολών μπορούμε να τροποποιήσουμε παράμερους – ιδιότητες σε ένα Source :

```
void alSource{n}{if}{v} (ALuint sourceName, ALenum paramName, T *values);
```

Αντίστοιχα μπορούμε να αντλήσουμε τις πληροφορίες αυτές, χρησιμοποιώντας τους αντίστοιχους τύπους εντολών :

```
void alGetSource{n}{if}{v} (ALuint sourceName, ALenum paramName, T *values);
```

Εκτός από τις τρεις παραπάνω βασικές ιδιότητες, το αντικείμενο *Source* διαθέτει μια πληθώρα ιδιοτήτων, που ρυθμίζουν διάφορες παραμέτρους του αντικειμένου και χρησιμοποιούν τους αντιστοίχους τύπους που φαίνονται παραπάνω. [35]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_SOURCE_RELATIVE	AL_TRUE AL_FALSE	AL_FALSE

Όταν το AL_SOURCE_RELATIVE τεθεί AL_TRUE δείχνει ότι η θέση, η ταχύτητα, ο κώνος και οι ιδιότητες κατεύθυνσης ενός *Source* πρόκειται να ερμηνευθούν σχετικά με τη θέση του *Listener*. [35]

```
alSourcei(Sources, AL_SOURCE_RELATIVE, AL_TRUE);
```

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_SOURCE_TYPE	AL_UNDETERMINED AL_STATIC AL_STREAMING	AL_UNDETERMINED

Η AL_SOURCE_TYPE είναι μια λειτουργία ανάγνωσης, που δείχνει εάν ένα *Source* είναι έτοιμο να περιμένει στη σειρά τους buffers ή είναι έτοιμο να χρησιμοποιήσει έναν static buffer ή είναι σε μια ακαθόριστη διαδικασία, όπου μπορεί να χρησιμοποιηθεί είτε για τη ροή, είτε για την αναπαραγωγή ήχου. [35]

Όταν ένα *Source* δημιουργείται, τοποθετείται σε κατάσταση AL_UNDETERMINED. Όταν ένας buffer τοποθετηθεί πάνω σε ένα *Source,* η κατάστασή του μεταβάλλεται σε AL_STATIC. Αν όμως ο buffer για να τοποθετηθεί χρησιμοποίησε την συνάρτηση alSourceQueueBuffers, τότε η κατάστασή του μεταβάλλεται σε AL_STREAMING. Αν τοποθετηθεί η τιμή NULL, αντί για buffer θα προκληθεί μια επαναρίθμηση του *Source* στο state AL_UNDETERMINED και η ένωση οποιουδήποτε buffer με ένα *Source* ροής θα αλλάξει σε AL_STATIC. Η προσπάθεια να τοποθετηθεί στην ουρά ένας buffer σε έναν static source, θα οδηγήσει σε ένα λάθος AL_INVALID_OPERATION. [35]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_BUFFER	Οποιοδήποτε bufferName	AL_NONE

Το AL_BUFFER διευκρινίζει το τρέχον buffer object, που θέλει να συνδεθεί στην είσοδο στην ουρά αναμονής ενός Source. Επομένως όταν θέλουμε να συνδέσουμε ένα buffer σε μια πηγή, μπορούμε να χρησιμοποιήσουμε την παρακάτω γραμμή κώδικα. [35]

```
alSourcei (Sources, AL_BUFFER, Buffer);
```

Η χρησιμοποίηση του AL_BUFFER σε ένα Source που έχει τεθεί σε κατάσταση AL_STOPPED ή AL_INITIAL, εκκενώνει ολόκληρη την ουρά αναμονής και κατόπιν συνδέει τον buffer που διευκρινίζεται. Ενώ για ένα Source που έχει τεθεί σε κατάσταση AL_PLAYING ή AL_PAUSED, η ρύθμιση AL_BUFFER θα οδηγήσει σε ένα λάθος AL_INVALID_OPERATION. Το AL_BUFFER μπορεί να εφαρμοστεί μόνο όταν ο Source είναι σε κατάσταση AL_INITIAL και AL_STOPPED. [35]

Ένα άκυρο ονόμα buffer (είτε επειδή το όνομα του buffer δεν υπάρχει, είτε επειδή εκείνος ο buffer δεν μπορεί να συνδεθεί με το διευκρινισμένο Source) θα οδηγήσει σε ένα λάθος AL_INVALID_VALUE. Αντίστοιχα ένα άκυρο όνομα Source θα οδηγήσει σε ένα λάθος AL_INVALID_NAME. [35]

Η παράμετρος AL_NONE (NULL ή 0), είναι ένα έγκυρο όνομα buffer. [35]

```
alSourcei(Sources, AL_BUFFER, AL_NONE);
```

Η λειτουργία που φαίνεται παραπάνω είναι ένας έγκυρος τρόπος απελευθέρωσης της τρέχουσας ουράς αναμονής σε ένα Source, που είναι σε κατάσταση AL_INITIAL ή AL_STOPPED. Αντίθετα όταν ο Source είναι σε κατάσταση AL_PLAYING ή AL_PAUSED θα προκληθεί ένα λάθος AL_INVALID_OPERATION [35]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_LOOPING	AL_TRUE AL_FALSE	AL_FALSE

Το AL_LOOPING είναι ένα flag που δείχνει ότι ο προκαθορισμένος Source δεν θα μπει σε κατάσταση AL_STOPPED μόλις φθάσει στο τέλος του τελευταίου buffer. Άντ' αυτού, ο Source θα μετατραπεί αμέσως σε AL_INITIAL και AL_PLAYING. Το AL_LOOPING μπορεί να αλλάξει σε μια πηγή σε οποιοδήποτε κατάσταση και αν βρίσκεται και προφανώς κατά την κατάσταση AL_PLAYING. [35]

```
alSourcei(Sources, AL_LOOPING, AL_TRUE);
```

Το OpenAL API δεν διευκρινίζει έναν ενσωματωμένο streaming μηχανισμό. Δεν υπάρχει κανένας μηχανισμός για την τοποθέτηση stream data σε ένα buffer object. Άντ' αυτού, το API διαθέτει έναν πιο εύκαμπτο και ευπροσάρμοστο μηχανισμό τοποθετώντας σε ουρά αναμονής τους buffers στους sources. Υπάρχουν πολλοί τρόποι να χρησιμοποιηθεί αυτό το χαρακτηριστικό γνώρισμα, δημιουργώντας streaming μέσα από αυτό. [35]

Το streaming αντικαθίσταται από static buffers που τοποθετούνται σε ουρά αναμονής. Αυτό κάνει αποτελεσματική οποιαδήποτε ενέργεια εναποθήκευσης στην προσωρινή μνήμη multi-buffer στην εφαρμογή, ενώ ταυτόχρονα επιτρέπει στην εφαρμογή να επιλέξει πόσους buffers θέλει να χρησιμοποιήσει, το μέγεθός τους και εάν αυτοί επαναχρησιμοποιούνται. Οι Buffers μπορούν να τοποθετηθούν στην ουρά αναμονής και αφότου χρησιμοποιηθούν μπορούν να διαγραφούν και εντέλει να ξαναγεμίσουν και να ξανατοποθετηθούν στην ουρά. Δεν υπάρχει κανένας περιορισμός. [35]

Το Looping μπορεί να εφαρμοστεί ώστε να επαναλαμβάνει τους buffers στην ουρά αναμονής. Άπειροι βρόχοι μπορούν (θεωρητικά) να ολοκληρωθούν. Ο διαχωρισμός των μεγάλων samples σε διαφορετικούς buffers, που διατηρούνται σε μια ουρά αναμονής, έχει ευδιάκριτα πλεονεκτήματα και συνιστάται από την OpenAL. Επίσης συνιστάται το μόνιμο loop ενός buffer να χρησιμοποιεί την συγκεκριμένη διαδικασία. [35]

Μια εφαρμογή μπορεί να περιμένει στην ουρά αναμονής ένα ή πολλαπλά ονόματα buffers, χρησιμοποιώντας την alSourceQueueBuffers. Οι buffers θα τοποθετηθούν στην ουρά αναμονής, σύμφωνα με την σειρά που εμφανίζονται στη ουρά. [35]

```
void alSourceQueueBuffers (ALuint sourceName, ALsizei numBuffers,
                           ALuint *bufferNames);
```

Αυτή η εντολή μπορεί να χρησιμοποιηθεί σε οποιαδήποτε κατάσταση και αν βρίσκεται ένα source. Για να επιτρέψει ένα streaming η ουρά αναμονής πρέπει να είναι δυνατή σε ένα Source AL_PLAYING. Όλοι οι buffers σε μια ουρά αναμονής πρέπει να έχουν το ίδιο format και τις ίδιες ιδιότητες, με εξαίρεση των NULL buffer, που μπορεί πάντα να τοποθετηθεί στη ουρά αναμονής. Μια προσπάθεια να αναμιχθούν τα formats ή άλλες ιδιότητες buffer θα οδηγήσει σε μια αποτυχία και σε ένα λάθος AL_INVALID_VALUE. Εάν η λειτουργία της ουράς αναμονής αποτύχει, η ουρά αναμονής του source θα παραμείνει αμετάβλητη (ακόμα κι αν μερικοί από τους buffer μπορούσαν να συνδεθούν στην ουρά). [35]

Όταν τοποθετηθούν στην ουρά αναμονής οι buffers και ενώ είναι εκκρεμής η επεξεργασία τους, δεν πρέπει να αλλάξουν. Η αφαίρεση μιας δεδομένης ουράς αναμονής δεν είναι δυνατή, εκτός αν το Source είναι σταματημένο ή εάν η καταχωρημένη ουρά αναμονής έχει υποβληθεί ήδη σε επεξεργασία (το Source σε κατάσταση AL_PLAYING ή AL_PAUSED, αλλά έχει ήδη εκτελέσει την συγκεκριμένη ουρά). Ένα AL_PLAYING source θα μπει σε κατάσταση AL_STOPPED όταν ολοκληρώσει την αναπαραγωγή ήχου του τελευταίου buffer της ουράς αναμονής. Η συμπεριφορά είναι ίδια με την σύνδεση ενός ενιαίου buffer σε ένα source. [35]

Η εντολή alSourceUnqueueBuffers αφαιρεί τον αριθμό των καταχωρημένων buffers που έχουν τελειώσει την επεξεργασία από την ουρά αναμονής. [35]

```
void alSourceUnqueueBuffers (ALuint sourceName, ALsizei numBuffers,
                             ALuint *bufferNames);
```

Αν ζητούνται περισσότεροι buffers από αυτούς που είναι διαθέσιμοι, η λειτουργία θα αποτύχει και θα προκληθεί ένα λάθος AL_INVALID_VALUE. [35]

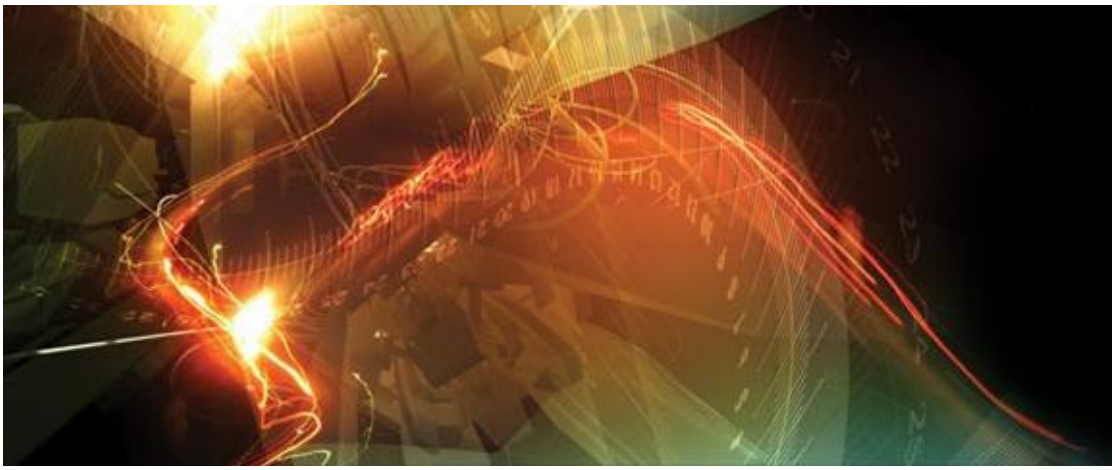
Η OpenAL διαθέτει και δύο καταστάσεις ερώτησης για την ουρά αναμονής:

Ερώτηση για τον αριθμό των buffers στην ουρά αναμονής ενός δεδομένου source. Αυτό περιλαμβάνει εκείνους που δεν έχουν παιχτεί ακόμα, ένα που παίζεται αυτήν την περίοδο και αυτούς που έχουν ήδη παιχτεί . [35]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_BUFFERS_QUEUED	[0, Οποιαδήποτε]	Καμία

Ερώτηση για τον αριθμό των buffers που έχουν παιχτεί από ένα δεδομένου source. Έμμεσα, η ερώτηση αυτή δίνει τον buffer που παίζει αυτήν την περίοδο.[35]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_BUFFERS_PROCESSED	[0, Οποιαδήποτε]	Καμία



Εικόνα 16 Virtual World

Εξ ορισμού, η OpenAL χρησιμοποιεί τα δευτερόλεπτα και τα Hertz ως μονάδες για το χρόνο και τη συχνότητα αντίστοιχα. Για τη συχνότητα, η βασική μονάδα είναι 1/second ή Hertz. Η συχνότητα των samples και η συχνότητα cut-offs ή άλλοι παράμετροι φίλτρων που διευκρινίζουν τις συχνότητες, εκφράζονται στις μονάδες των Hertz. [35]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_PITCH	[0.0f, Οποιαδήποτε]	1.0f

Το AL_PITCH διευκρινίζει το επιθυμητό pitch shift, η τιμή 1.0 είναι ίση με μηδενική μετατόπιση, ενώ κάθε διπλασιασμός είναι ίσος με ένα pitch shift 12 (αύξηση μιας οκτάβας). [35]

```
alSourcei(Sources, AL_PITCH, 1.0);
```

Μία ακόμα παράμετρος που μπορεί να ρυθμιστεί μέσα από το source αντικείμενο είναι το offset. [35]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT	ΠΕΡΙΓΡΑΦΗ
AL_SEC_OFFSET	[0.0f, Οποιαδήποτε]	-	Το offset του ήχου, που εκφράζεται σε δευτερόλεπτα.
AL_SAMPLE_OFFSET	[0.0f, Οποιαδήποτε]	-	Το offset του ήχου, που εκφράζεται σε δείγματα -sample
AL_BYTE_OFFSET	[0.0f, Οποιαδήποτε]	-	Το offset του ήχου, που εκφράζεται σε ψηφία –byte

4.5 Listener

Το *Listener* object καθορίζει διάφορες ιδιότητες που αντιπροσωπεύσουν τον τελικό ήχο που θα ακούσει ο χρήστης. Όπως έχουμε προαναφέρει, είναι μοναδικός για κάθε OpenAL Context και δεν έχει κανένα όνομα. [35]

Με τον έλεγχο του listener η εφαρμογή ελέγχει τον τρόπο που ο χρήστης συμπεριφέρεται σε ένα εικονικό 3D κόσμο, επειδή ο listener καθορίζει των προσανατολισμό και άλλες παραμέτρους που επηρεάζουν την έξοδο. Οι βασικές ιδιότητες του *Listener* είναι ίδιες με αυτές του *Source:* AL_POSITION, AL_VELOCITY, AL_GAIN. [35]

Επιπλέον, παρέχεται μια ακόμα κρίσιμη ιδιότητα, η AL_ORIENTATION. Πριν όμως αναλύσουμε την συγκεκριμένη ιδιότητα, πρέπει να κατανοήσουμε τι πραγματικά είναι το Listener object. Ουσιαστικά είναι ένα αντικείμενο της OpenAL, που προσπαθεί να προσομοιώσει τον χρήστη και να τον εισάγει σε έναν εικονικό ακουστικό χώρο. Θα μπορούσαμε να πούμε ότι είναι το αυτί του ακροατή. [35]

Ένας συνηθισμένος χρήστης – ακροατής, συνήθως βρίσκεται μπροστά από έναν τυποποιημένο υπολογιστή, σε μια συνηθισμένη θέση, με δύο stereo ηχεία. Όταν ο χρήστης αυτός εκτελεί μια 3D εφαρμογή, πρέπει ακουστικά να μπει «μέσα στην οθόνη», ώστε να μπορεί να νιώσει ότι βρίσκεται μέσα σε αυτόν τον τρισδιάστατο κόσμο που παρακολουθεί. Ο Listener είναι αυτός που πρέπει να βάλει τον χρήστη «μέσα στην Οθόνη». Αυτή η διαδικασία αποτελεί το πιο κρίσιμο σημείο της λειτουργίας του συγκεκριμένου αντικειμένου. [35 49 ,50]

Εικόνα 17 The Listener's Ears

Για να κατανοήσουμε καλύτερα την λειτουργία του αντικειμένου αυτού, θα χρησιμοποιήσουμε ένα παράδειγμα και θα μεταφερθούμε στον τρισδιάστατο κόσμο:

Έστω ότι ο χρήστης ξεκινάει μια 3D εφαρμογή και για την ακρίβεια ξεκινάει ένα εικονικό παιχνίδι, που έχει πλήρη δυνατότητα μετακίνησης του εικονικού του χαρακτήρα, σαν να ήταν στον «πραγματικό κόσμο». Ο εικονικός αυτός χαρακτήρας για τον ήχο προσομοιώνεται με το αντικείμενο του Listener. Η default θέση, δηλαδή το σημείο που θα ξεκινήσει ο χαρακτήρας για τα περισσότερα παιχνίδια, είναι σε *όρθια* θέση, ενώ επίσης πάντα βλέπει *μπροστά*. [35 49 ,50, 47,51,53]

Εικόνα 18 Default θέση κατά το Load ενός παιχνιδιού.

Αν ο χρήστης ζητήσει από τον χαρακτήρα του να μετακινηθεί προς τα εμπρός, μαζί με το γραφικό περιβάλλον του χαρακτήρα που θα «μετακινηθεί», θα «μετακινηθεί» και το ακουστικό περιβάλλον. Ακουστικά, αυτό που θα αλλάξει θα είναι το Listener AL_POSITION. Αν ήταν {X =0, Y=0, Z=0} και έχει ζητηθεί από τον χαρακτήρα να μετακινηθεί κατά Z=-10 τότε το Listener AL_POSITION θα είναι {X=0, Y=0, Z=-10}. 35 49 ,50, 47,51,53

Έστω ότι στο σημείο Listener AL_POSITION {X =0, Y=0 ,Z=10} ο χρήστης ακούει έναν ήχο από πίσω του στην θέση Sources AL_POSITION {X =0, Y=0 ,Z=5} και ακούγοντας τον κάνει μια περιστροφή 180^{o}. Σε αυτήν την περίπτωση θα υπάρχει αλλαγή *κατεύθυνσης*, κάτι που δεν μπορεί να περιγραφεί από το AL_POSITION του χαρακτήρα. Κάνοντας αυτήν την περιστροφή ουσιαστικά όλοι οι ήχοι που βρίσκονταν πίσω πρέπει να μεταφερθούν μπροστά, ενώ όλοι οι ήχοι που ακούγονταν δεξιά πρέπει να μεταφερθούν αριστερά. 35 49 ,50, 47,51,53

Ο εικονικός κόσμος, σε πολλά παιχνίδια, παρέχει και μια ακόμα δυνατότητα στον χρήστη που τον αποδεσμεύει από τον χαρακτήρα του. Η αποδέσμευση αυτή επιτυγχάνεται μέσω της κίνησης της κάμερας του, κάτι που για τις νέες εφαρμογές αποτελεί ένα standard. Ουσιαστικά ο Listener σε αυτές τις εφαρμογές είναι η κάμερα. Ο χρήστης μπορεί να ακολουθεί τον χαρακτήρα του και σε γενικές γραμμές αυτό κάνει, αλλά μπορεί να δει και πράγματα που δεν μπορεί να δει ο χαρακτήρας του. Ταυτόχρονα προφανώς μεταβάλλεται και το ακουστικό του περιβάλλον. 35 49 ,50, 47,51,53

Επομένως ο Listener είναι η κάμερα του χρήστη που:

- Σε ορισμένες περιπτώσεις γίνεται ένα με τον χαρακτήρα. Αυτό συμβαίνει σε παιχνίδια που ο χρήστης βλέπει μόνο τα χέρια του χαρακτήρα του. 35 49 ,50, 47,51,53

- Σε άλλες περιπτώσεις ακολουθεί τον χαρακτήρα, έχοντας όμως την δυνατότητα να δει μπροστά \ πίσω, δεξιά \ αριστερά, να κάνει zoom in \ out και γενικά να ελέγχει ένα τρισδιάστατο άξονα γύρω από τον χαρακτήρα του. 35 49 ,50, 47,51,53

- Σε πιο πρόσφατες εφαρμογές, ο χρήστης μπορεί να χειριστεί ταυτόχρονα πολλαπλούς χαρακτήρες και να βλέπει ή να ακολουθεί όποιον από τους χαρακτήρες του επιθυμεί (π.χ Sims 2: ο χρήστης ελέγχει μια ολόκληρη οικογένεια και μπορεί να ακολουθεί όποιον από τους χαρακτήρες της οικογένειας επιθυμεί). 35 49 ,50, 47,51,53

Είναι προφανές ότι Listener και οι ιδιότητές του εξαρτώνται άμεσα από το τι θέλει να κάνει μια τρισδιάστατη εφαρμογή και το πως παρουσιάζει στον χρήστη τον εικονικό της κόσμο. Η κάμερα είναι ο Listener και επομένως το ηχητικό περιβάλλον αλλάζει σε συνάρτηση με την κίνησή της. 35 49 ,50, 47,51,53

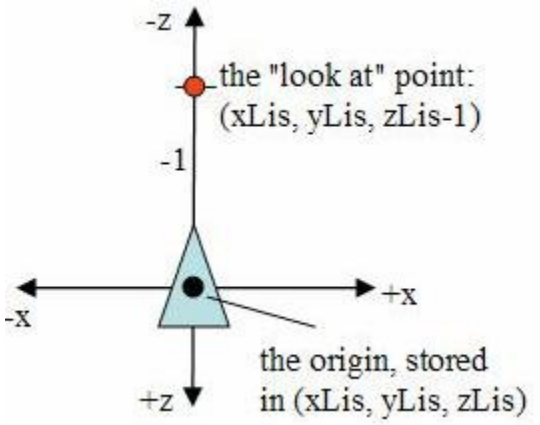
Εικόνα 19 The Elder Scrolls IV: Oblivion είναι ένα από τα παιχνίδια που παρέχει δύο δυνατότητες στους χρήστες.

Στην OpenAL η κίνηση του Listener περιγράφεται αρχικά από το AL_POSITION που, όπως έχουμε προαναφέρει, είναι η θέση της κάμερας στον τρισδιάστατο κόσμο. Για την περιγραφή την κατεύθυνσης χρησιμοποιεί την ιδιότητα AL_ORIENTATION. Όλες αυτές οι λειτουργίες προσανατολισμού επιτυγχάνονται μέσω της συγκεκριμένης ιδιότητας: [35]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT	ΠΕΡΙΓΡΑΦΗ
AL_ORIENTATION	Οποιαδήποτε αριθμητική τιμή εκτός από NaN	{(0.0f, 0.0f, -1.0f), (0.0f, 1.0f, 0.0f)}	Δείχνει τον προσανατολισμό των ακροατών. Είναι ένα ζεύγος τριπλών στοιχείων –διανυσμάτων.

Η AL_ORIENTATION αποτελείται από έξι τιμές. Είναι ένα ζεύγος τριπλών στοιχείων από δύο διανύσματα, που εξετάζουν τον προσανατολισμό του χρήστη. Η OpenAL αναμένει ότι τα δυο διανύσματα είναι γραμμικώς ανεξάρτητα. Αν τα δύο διανύσματα είναι γραμμικώς εξαρτημένα, η συμπεριφορά είναι απροσδιόριστη. [35]

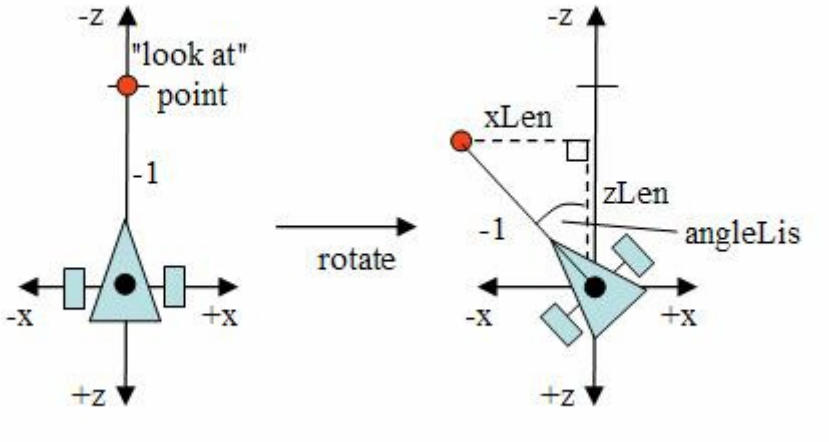
Οι πρώτες τρεις τιμές αναφέρονται στο “Look at point”(X, Y, Z). Το “Look at point” είναι το σημείο που «βλέπει» ο χρήστης. Είναι το σημείο που βλέπουν τα μάτια του χρήστη, δηλαδή το σημείο view της κάμερας. [35 49 ,50,52,54,51,55,56,53]



Εικόνα 20 The Listener's Orientation

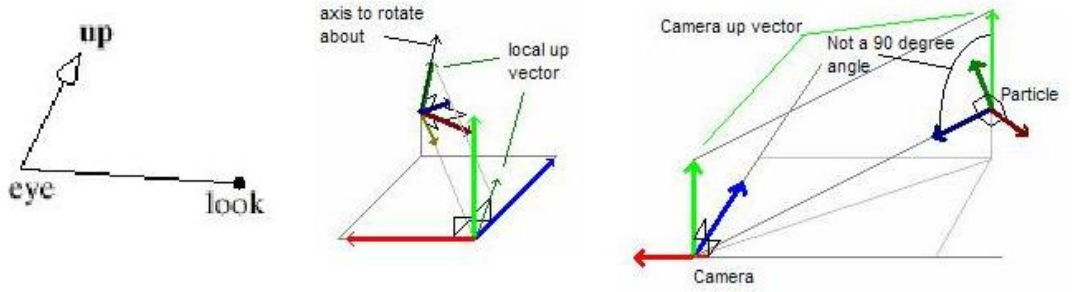
Αν παραδείγματος χάριν, θέλαμε να επιτύχουμε την περιστροφή του χρήστη κατά 180^{o}, αυτό που θα χρειαζότανε να κάνουμε είναι να αλλάξουμε το πρώτο διάνυσμα. Η

default τιμή του είναι (0.0f, 0.0f, -1.0f). Για μια περιστροφή 180^{o}, θα έπρεπε απλά να τοποθετήσουμε στον άξονα Z την τιμή 1.0f. Επομένως το πρώτο διάνυσμα θα είχε την μορφή (0.0f, 0.0f, 1.0f). 35 49 ,50,52,54,51,55,56,53



Εικόνα 21 Rotating the Listener and the "look at" Point

Το δεύτερο διάνυσμα της ιδιότητας AL_ORIENTATION, είναι το "Up". Είναι ένα διάνυσμα που περιγράφεται ιδιαίτερα δύσκολα και έχει άμεση σχέση με τον τρόπου που λειτουργεί η κάμερα στην OpenGL. Θα μπορούσαμε να πούμε ότι είναι το ανώτατο σημείο προβολής της κάμερας. Ηχητικά προσομοιώνει την κίνηση πάνω –κάτω. Για παράδειγμα, αν σκεφτούμε ότι είμαστε ακίνητοι σε ένα σημείο και κουνάμε μόνο το κεφάλι μας πάνω κάτω, είναι σίγουρο ότι θα υπάρχει μετατόπιση στο ηχητικό ερέθισμα, αν και η μετατόπιση αυτή δεν θα είναι τόσο αισθητή. Το συγκεκριμένο διάνυσμα περιγράφει ακριβώς αυτήν την κίνηση, αλλά για την κάμερα, παίρνοντας ως βάση το πάνω σημείο της μετατόπισης αυτής. Αυτό το σημείο δεν επηρεάζει ιδιαίτερα το ηχητικό αποτέλεσμα, παρά μόνο σε περιπτώσεις που συνδυάζεται με όλα τα άλλα και βασικότερα με την κίνηση μιας κάμερας. 35 49 ,50,52,54,51,55,56,53



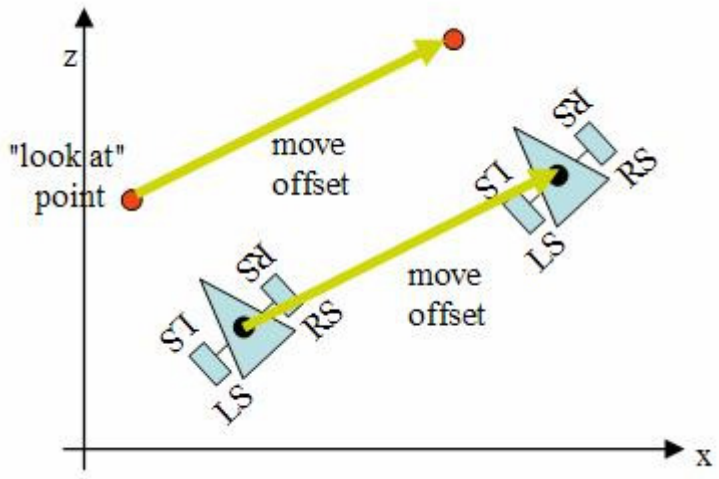
Εικόνα 22 Up Vector

Οι default τιμές της ιδιότητας AL_ORIENTATION είναι ίδιες με τις τιμές που τοποθετούνται και στην OpenGL ως default για τα αντίστοιχα σημεία της κάμερας.

Επομένως όταν αρχικοποιούμε έναν τρισδιάστατο ακουστικό χώρο στον κώδικά μας, θα μπορούσαμε να συμπεριλάβουμε τις παρακάτω αρχικοποιήσεις για το αντικείμενο του Listener. [35 49, 50, 52 54, 51, 55,56]

```
//Position of the Listener.
ALfloat ListenerPos[] = { 0.0, 0.0, 0.0 };

// Velocity of the Listener.
ALfloat ListenerVel[] = { 0.0, 0.0, 0.0 };

//Orientation of the Listener.
//(first 3 elements are "at", second 3 are "up")
ALfloat ListenerOri[] = { 0.0, 0.0, -1.0,  0.0, 1.0, 0.0 };

void SetListenerValues()
{
      alListenerfv(AL_POSITION,    ListenerPos);
      alListenerfv(AL_VELOCITY,    ListenerVel);
      alListenerfv(AL_ORIENTATION, ListenerOri);
}
```

Η αλλαγή της θέσης του Listener απαιτεί επίσης μια αναπροσαρμογή στον προσανατολισμό του, όπως φαίνεται στο σχήμα που ακολουθεί. [35 49, 50, 52 54, 51, 55,56]



Εικόνα 23 Moving the Listener and its “look at” Point

Η αλλαγή της θέσης του Listener περνάει μέσα από πολλά στάδια, πριν φτάσει στην OpenAL. Κατ’ αρχήν, μέσω του GUI (Graphical user Interface), η εφαρμογή «συλλέγει» τις κινήσεις του ποντικιού και του πληκτρολογίου και τις μεταφέρει στο OpenGL Matrix, μέσω εντολών της OpenGL. Το matrix είναι ο τρόπος που χρησιμοποιεί η

OpenGL για να διευκρινίσει τα διανύσματα *ν* διαστάσεων και να τα μετασχηματίσει σε ένα κατάλληλο πακέτο. Το πακέτο αυτό έχει την μορφή που φαίνεται παρακάτω: [35,49,50,52,54,51,55,56,53]

4x4 Matrix

[A1 A2 A3 A4]
[B1 B2 B3 B4]
[C1 C2 C3 C4]
[D1 D2 D3 D4]

Στην C++ ένα Matrix μπορεί να είναι της μορφής Matrix[4][4] ή Matrix[16]. Το Matrix διαθέτει διάφορα Mode. Αυτό που ασχολείται με την κίνηση είναι το GL_MODELVIEW_MATRIX. [35,49,50,52,54,51,55,56,53]

Στην συνεχεία η Openal ανασύρει τις πληροφορίες που την ενδιαφέρουν από το MODELVIEW MATRIX και τις τοποθετεί στον Listener. Αυτή η διαδικασία γίνεται μέσα σε μία function, που θα μπορούσαμε να την ονομάσουμε Scene. Η function αυτή ανανεώνεται συνέχεια και ενημερώνεται για αλλαγες στις τιμές μέσω του GUI, με αποτελεσμα να υπάρχει μια συνεχής αλληλεπίδραση. Οι εσωτερικές function της Scene για τον Lisener, θα μπορούσαν να είναι της μορφής που φαίνεται παρακάτω. [35,49,50,52,54,51,55,56,53]

```
void SetListenerPosition(float x, float y, float z)
{
	//set the position using 3 seperate floats
	alListener3f(AL_POSITION, x,y,z);
}

void SetListenerOrientation(float fx, float fy, float fz, float ux,
float uy, float uz)
{
	//set the orientation using an array of floats
	float vec[6];
	vec[0] = fx;
	vec[1] = fy;
	vec[2] = fz;
	vec[3] = ux;
	vec[4] = uy;
	vec[5] = uz;
	alListenerfv(AL_ORIENTATION, vec);
}
```

Ο προγραμματισμός της λειτουργίας αυτής στο σύνολό της αποτελεί μια ιδιαίτερα επίπονη και αργή διαδικασία και σε πολλές περιπτώσεις έχουν συμβεί σοβαρά λάθη. Δεν

είναι λίγα τα παιχνίδια που δεν παίχτηκαν ποτέ, μόνο και μόνο γιατί η κάμερά τους δεν ήταν εύχρηστη και δεν έδινε στον χρήστη την αίσθηση που έχει συνηθίσει να έχει. Επομένως είναι παρά πολύ σημαντικό ο προγραμματιστής να γνωρίζει καλά όλη την λειτουργία, ώστε να μπορέσει να δώσει τις σωστές τιμές στην OpenGL και εντέλει να ανασύρει τις σωστές τιμές για την OpenAL από το Matrix. [35,49 ,50,52,54,51,55,56,53]

Εκτός όμως από τον κόσμο των παιχνιδιών, η λειτουργία αυτή μπορεί να γίνει ακόμα πιο πολύπλοκη στα ερευνητικά συστήματα που παρουσιάζονται και ονομάζονται virtual reality η αλλιώς εικονική πραγματικότητα [57]

Εικόνα 24 virtual reality

4.6 Επιπρόσθετες Υλοποιήσεις και Δυνατότητες

Η OpenAL παρέχει τα μέσα για την φυσική προσομοίωση του ήχου, σύμφωνα με την απόσταση, καθώς και για την αύξηση ή μείωση αυτής της επίδρασης. Δεν καθορίζει τις μονάδες μέτρησης για την απόσταση. Η εφαρμογή είναι ελεύθερη να χρησιμοποιήσει τις μονάδες μέτρησης που επιθυμεί. Εντούτοις, τα προκύπτοντα αποτελέσματα δεν εξαρτώνται από τις μονάδες μέτρησης της απόστασης που χρησιμοποιείται από την εφαρμογή για να εκφραστούν οι συντεταγμένες του source και listener. Η εφαρμογή είναι αυτή που πρέπει να ρυθμίσει το source gain για να επιτύχει το επιθυμητό αποτέλεσμα. Οι προδιαγραφές της OpenAL επιβάλουν τον ευκλείδειο υπολογισμό των αποστάσεων. [35,36]

Η OpenAL αυτήν την περίοδο υποστηρίζει τρεις τρόπους λειτουργίας-πρότυπα, όσον αφορά τη μείωση απόστασης. Η εφαρμογή μπορεί να επιλέξει ανά context, ένα από αυτά τα πρότυπα ή να επιλέξει να θέσει εκτός λειτουργίας την εξαρτώμενη μείωση απόστασης. [35,36]

```
void alDistanceModel(ALenum modelName);
```

Distance models	
AL_NONE	Θέτει εκτός λειτουργίας, παρακάμπτει τον υπολογισμό μείωσης απόστασης για όλα τα sources
AL_INVERSE_DISTANCE	Πρότυπο αντίστροφης απόστασης
AL_INVERSE_DISTANCE_CLAMPED	Σταθερό αντίστροφο πρότυπο απόστασης
AL_LINEAR_DISTANCE	Γραμμικό πρότυπο απόστασης
AL_LINEAR_DISTANCE_CLAMPED	Σταθερό γραμμικό πρότυπο απόστασης
AL_EXPONENT_DISTANCE	Εκθετικό πρότυπο απόστασης
AL_EXPONENT_DISTANCE_CLAMPED	Σταθερό εκθετικό πρότυπο απόστασης

Το προεπιλεγμένο πρότυπο είναι το AL_INVERSE_DISTANCE_CLAMPED. Το τρέχον πρότυπο απόστασης μπορεί να ερωτηθεί εκτελώντας την ακόλουθη γραμμή. [35,36]

```
void alGetIntegerv(AL_DISTANCE_MODEL, ALint *data);
```

Οι βασικές ιδιότητες που χρησιμοποιεί η OpenAL για τα 3 πρότυπα είναι : AL_REFERENCE_DISTANCE, AL_ROLLOFF_FACTOR , AL_MAX_DISTANCE. [35,36]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_REFERENCE_DISTANCE	[0, Οποιαδήποτε]	1.0f

Η παράμετρος αυτή χρησιμοποιείται για τον υπολογισμό της μείωσης απόστασης και βασίζεται στην αντίστροφη μείωση της απόστασης. Ανάλογα με το πρότυπο απόστασης θα ενεργήσει επίσης ως κατώτατο όριο απόστασης, κάτω από το οποίο το gain γίνεται σταθερό. [35,36]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_ROLLOFF_FACTOR	[0, Οποιαδήποτε]	1.0f

Η παράμετρος αυτή χρησιμοποιείται για τον υπολογισμό της μείωσης απόστασης και βασίζεται στην αντίστροφη μείωση της απόστασης. Για αποστάσεις μικρότερες από AL_MAX_DISTANCE και, ανάλογα με το πρότυπο απόστασης, μεγαλύτερες από AL_REFERENCE_DISTANCE, θα ανεβάσει τη αύξηση την απόστασης πέρα από την εφαρμόσιμη ακτίνα. Ειδικότερα, το AL_ROLLOFF_FACTOR μπορεί να τεθεί μηδέν για εκείνα τα sources που επιθυμείται να μην επηρεάζονται, σύμφωνα με την μείωση της απόστασης. [35,36]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_MAX_DISTANCE	[0, Οποιαδήποτε]	MAX_FLOAT

Η παράμετρος αυτή χρησιμοποιείται για τους υπολογισμούς μείωσης απόστασης, βασισμένους στην αντίστροφη απόσταση, αν χρησιμοποιείται το αντίστροφο σταθερό πρότυπο απόστασης. Σε αυτήν την περίπτωση αποστάσεις μεγαλύτερες από AL_MAX_DISTANCE θα μετατραπούν στο AL_MAX_DISTANCE. [35,36]

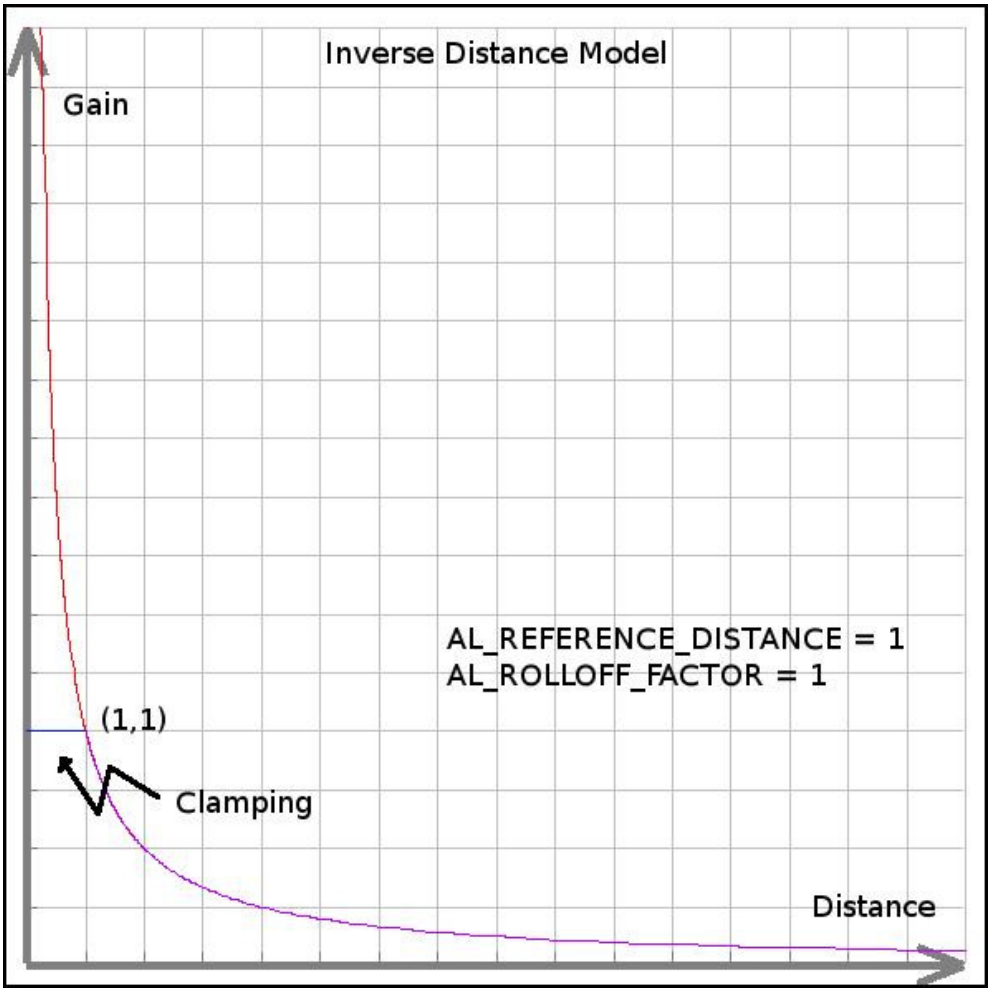
Η AL_MAX_DISTANCE εφαρμόζεται πριν την AL_MIN_GAIN. Έτσι εάν το gain της AL_MAX_DISTANCE είναι μεγαλύτερο από το AL_MIN_GAIN, τότε το AL_MIN_GAIN δεν θα έχει καμία επίδραση. [35,36]

Το πρότυπο AL_INVERSE_DISTANCE λειτουργεί σύμφωνα με τον ακόλουθο τύπο: [35,36]

```
gain = AL_REFERENCE_DISTANCE / (AL_REFERENCE_DISTANCE +
AL_ROLLOFF_FACTOR *(distance - AL_REFERENCE_DISTANCE));
```

Το πρότυπο AL_INVERSE_DISTANCE_CLAMPED λειτουργεί σύμφωνα με τον ακόλουθο τύπο: [35,36]

```
distance = max(distance, AL_REFERENCE_DISTANCE);
distance = min(distance, AL_MAX_DISTANCE);

gain = AL_REFERENCE_DISTANCE / (AL_REFERENCE_DISTANCE +
AL_ROLLOFF_FACTOR * (distance - AL_REFERENCE_DISTANCE));
```



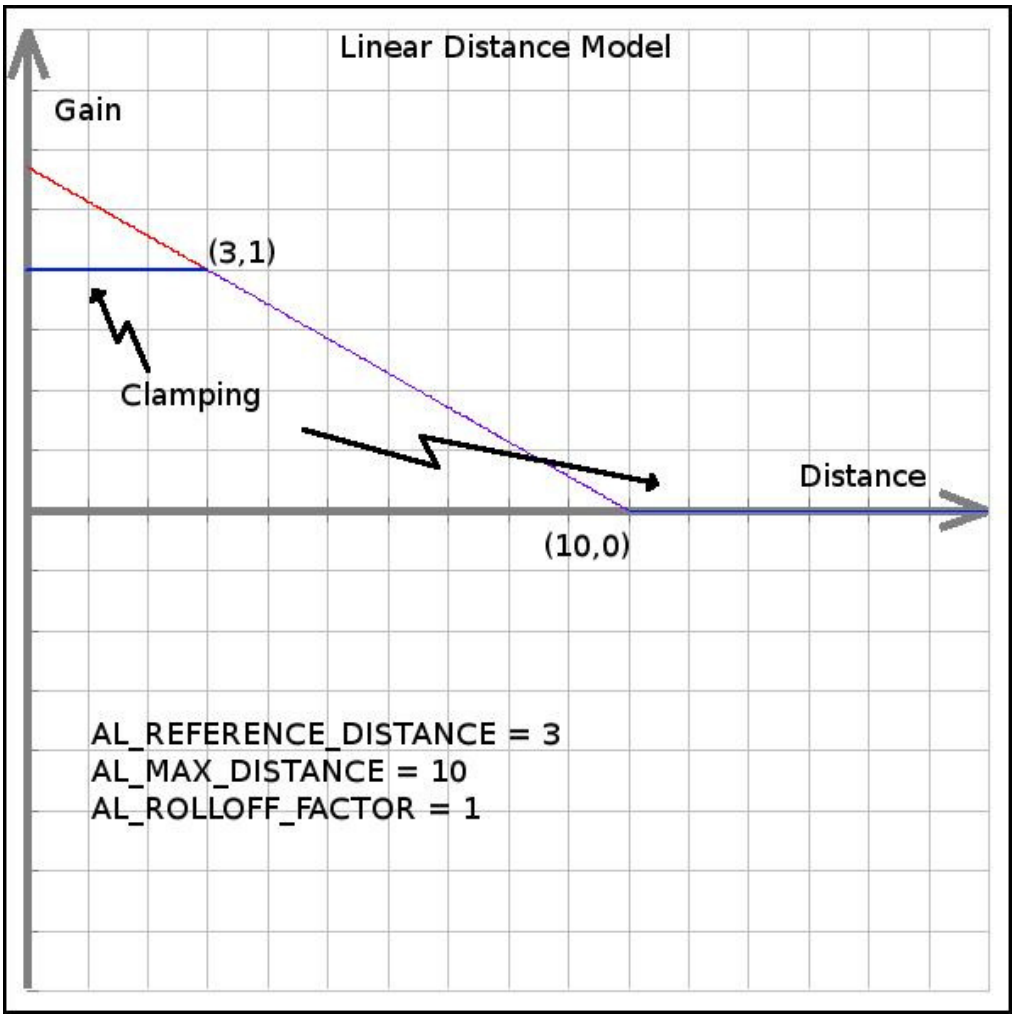
Εικόνα 25 Γραφική παράσταση που παρουσιάζει την αντίστροφη καμπύλη απόστασης[35,36]

Το πρότυπο AL_LINEAR_DISTANCE λειτουργεί σύμφωνα με τον ακόλουθο τύπο: [35,36]

```
distance = min(distance, AL_MAX_DISTANCE)

gain = (1 - AL_ROLLOFF_FACTOR * (distance -AL_REFERENCE_DISTANCE)/
       (AL_MAX_DISTANCE - AL_REFERENCE_DISTANCE))
```

Το πρότυπο AL_LINEAR_DISTANCE_CLAMPED λειτουργεί σύμφωνα με τον ακόλουθο τύπο: [35,36]

```
distance = max(distance, AL_REFERENCE_DISTANCE)
distance = min(distance, AL_MAX_DISTANCE)

gain = (1 - AL_ROLLOFF_FACTOR * (distance -AL_REFERENCE_DISTANCE) /
(AL_MAX_DISTANCE - AL_REFERENCE_DISTANCE))
```



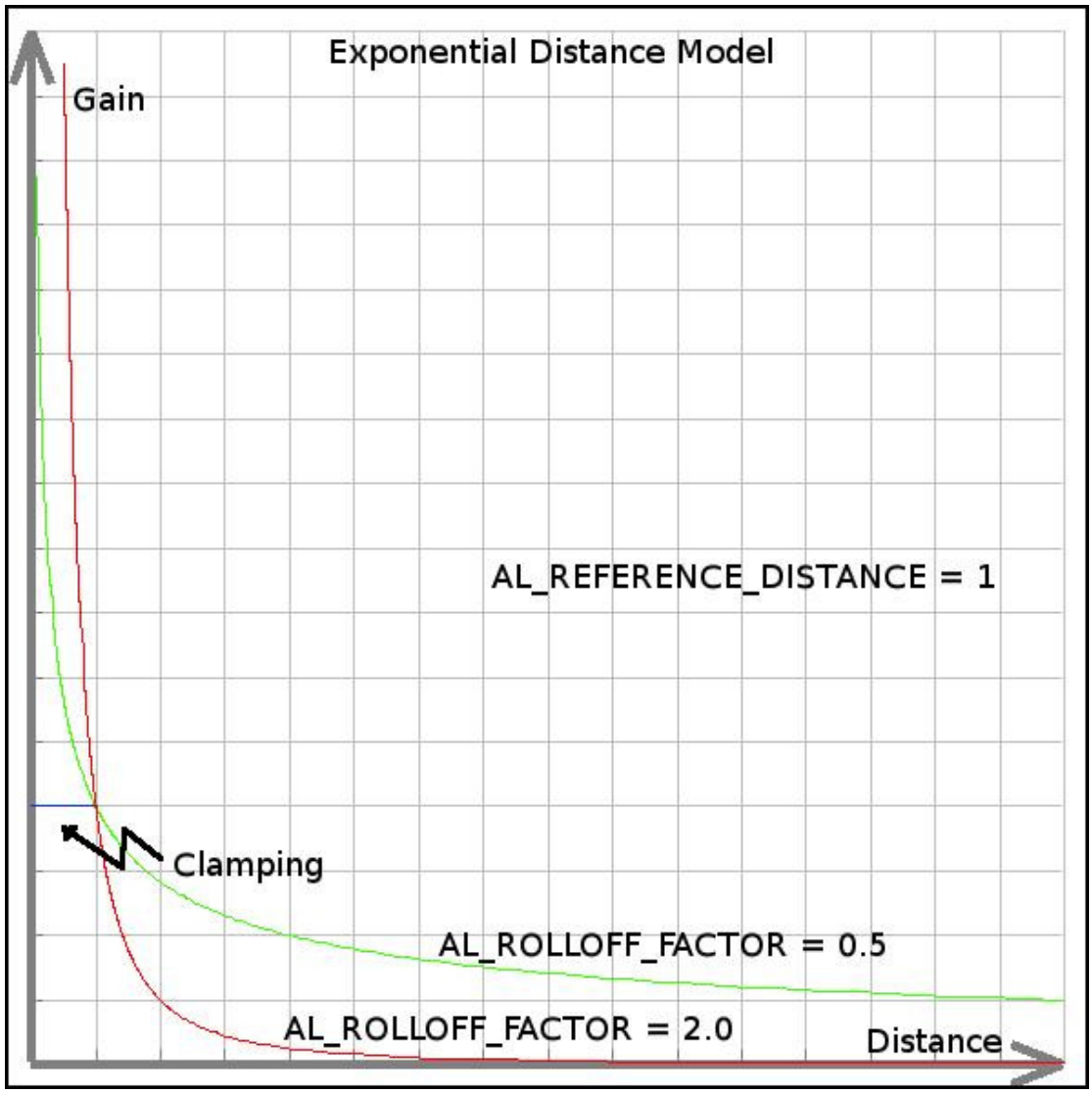
Εικόνα 26 Γραφική παράσταση που παρουσιάζει την γραμμική καμπύλη απόστασης. [35,36]

Το πρότυπο AL_EXPONENT_DISTANCE λειτουργεί σύμφωνα με τον ακόλουθο τύπο: [35,36]

```
gain = (distance / AL_REFERENCE_DISTANCE) ^ (- AL_ROLLOFF_FACTOR)
```

Το πρότυπο AL_EXPONENT_DISTANCE_CLAMPED λειτουργεί σύμφωνα με τον ακόλουθο τύπο: [35,36]

```
distance = max(distance, AL_REFERENCE_DISTANCE)
distance = min(distance, AL_MAX_DISTANCE)

gain = (distance / AL_REFERENCE_DISTANCE) ^(- AL_ROLLOFF_FACTOR)
```



Εικόνα 25 Γραφική παράσταση που παρουσιάζει την εκθετική καμπύλη απόστασης. [35,36]

Ένα επίσης απαραίτητο στοιχείο για την τοποθέτηση ενός ήχου στον τρισδιάστατο ηχητικό κόσμο είναι η κατεύθυνση. Η OpenAL για αυτόν τον λόγο παρέχει την ιδιότητα AL_DIRECTION. [35,36]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_DIRECTION	Οποιαδήποτε αριθμητική τιμή εκτός από NaN	(0.0f, 0.0f, 0.0f)

Εάν το AL_DIRECTION είναι ίσο με το μηδενικό διάνυσμα, το source δεν έχει συγκεκριμένη κατεύθυνση. Η τοποθέτηση ενός διανύσματος διαφορετικού από το μηδέν θα καταστήσει το source κατευθυντικό. Στο διάνυσμα αυτό δεν μπορούν να μετατραπούν ταυτόχρονα και οι τρεις τιμές, μιας και αυτό προσδιορίζει κίνηση του αντικειμένου και όχι κατεύθυνση. [35,36]

Συνοψίζοντας : Οι διαδικασίες που εκτελούνται-ακολουθούνται κατά την διαμόρφωση του gain είναι: [35]

- Πρώτα υπολογίζεται η μείωση απόστασης συμπεριλαμβάνοντας το ελάχιστο AL_REFERENCE_DISTANCE και το μέγιστο κατώτατο όριο AL_MAX_DISTANCE.
- Το αποτέλεσμα πολλαπλασιάζεται έπειτα με το source gain AL_GAIN.
- Εάν η πηγή είναι κατευθυντική και το AL_CONE_INNER_ANGLE είναι λιγότερο από το AL_CONE_OUTER_ANGLE, μια γωνία εξαρτώμενης μείωσης υπολογίζεται ανάλογα με το AL_CONE_OUTER_GAIN και πολλαπλασιάζεται με την εξαρτώμενη μείωση απόστασης. Ο προκύπτων παράγοντας μείωσης για την δεδομένη γωνία και την απόσταση μεταξύ του listener και της source, πολλαπλασιάζεται με την source AL_GAIN.

AL_CONE_INNER_ANGLE	Εσωτερική γωνία του ηχητικού κώνου σε μοίρες. Η προεπιλογή 360 σημαίνει ότι η εσωτερική γωνία καλύπτει ολόκληρο τον κόσμο και είναι ισοδύναμος με μια παντοκατευθυντική πηγή.Default: 360.0f
AL_CONE_OUTER_ANGLE	Εξωτερική γωνία του ηχητικού κώνου, σε μοίρες. Η προεπιλογή 360 σημαίνει ότι η εξωτερική γωνία καλύπτει ολόκληρο τον κόσμο. Εάν και η εσωτερική γωνία είναι επίσης 360, η ζώνη για την εξαρτημένη μείωση είναι μηδέν.Default: 360.0f
AL_CONE_OUTER_GAIN	Ο παράγοντας με τον οποίο το AL_GAIN πολλαπλασιάζεται για να καθορίσει το αποτελεσματικό κέρδος έξω από τον κώνο, που καθορίζεται από την εξωτερική γωνία. Values: [0.0f, 0.1f] Default: 0.0f

- Το αποτέλεσμα του gain συγκρίνεται με το κατώτατο και ανώτατο όριο AL_MIN_GAIN και AL_MAX_GAIN.
- Το αποτέλεσμα που είναι μεταξύ των [AL_MIN_GAIN, AL_MAX_GAIN], πολλαπλασιάζεται στη συνέχεια με το listener gain.

Το AL API διαθέτει επίσης ένα κατάλογο επεκτάσεων, που παρουσιάζονται αναλυτικά στο Κεφαλαίο 3: 6. Example: System info - Extensions.

Παραδείγματα ολοκληρωμένων υλοποιήσεων του AL API παρουσιάζονται στο ΚΕΦΑΛΑΙΟ 3, ενώ ένας πλήρης κατάλογος των λειτουργιών της AL παρουσιάζεται στο Παράρτημα 1:

ΠΙΝΑΚΑΣ 2.1: AL Functions
ΠΙΝΑΚΑΣ 2.2 AL Primitive Types
ΠΙΝΑΚΑΣ 2.3 AL Define Values

5. EFX API: Effects Extension

Αν και ο πυρήνας AL παρέχει διάφορες περίπλοκες τρισδιάστατες ακουστικές λειτουργίες για την χωροτοποθέτηση αντικειμένων, στερείται ορισμένα πολύ σημαντικά περιβαλλοντολογικά effects, όπως η αντήχηση, η ανάκλαση και η περίθλαση. Χωρίς αυτά τα περιβαλλοντολογικά effects, ένας ακροατής μπορεί να εντοπίσει την κατεύθυνση κάθε πηγής, αλλά δύσκολα θα αντιληφθεί πόσο μακριά είναι οι πηγές. Επίσης, ο ακροατής δεν έχει καμία ιδέα του περιβάλλοντος όπου βρίσκονται οι πηγές. Το EFX API (Effects Extensions) έρχεται να καλύψει αυτό το κενό με την προσθήκη διάφορων φίλτρων. [58]

Είναι προφανές ότι ένας ήχος θα ηχήσει τελείως διαφορετικά αν είναι τοποθετημένος σε ένα μικρό δωμάτιο απ΄ ότι σε ένα μεγάλο ναό. Αυτό που προφανώς αλλάζει είναι η αντήχηση του χώρου. Αν εξετάσουμε έναν ήχο που προέρχεται από το διπλανό δωμάτιο θα διαπιστώσουμε ότι η παρεμπόδιση του τοίχου δημιουργεί ένα «ποιοτικό κάλυμμα» στον ήχο. Χωρίς αυτές τις περιβαλλοντικές επιπτώσεις, ο ακροατής δεν μπορεί να επισημάνει πόσο μακριά είναι οι πηγές και δεν έχει καμία ιδέα του περιβάλλοντος όπου βρίσκονται οι πηγές. [58]

Ο περιβαλλοντικός ήχος σε μια τρισδιάστατη εφαρμογή παρέχει μια ευδιάκριτη ένδειξη των χαρακτηριστικών του τρισδιάστατου χώρου. Τα αποτελέσματα της αντήχησης μπορούν να παρέχουν στον ακροατή τα ηχητικά ερεθίσματα για να διακρίνει και να ενισχυθεί-διαμορφωθεί το συναίσθημα της χωροτοποθέτησης. [58]

Εικόνα 26 Εικονικά περιβάλλοντα που δημιουργήθηκαν στο περιβάλλον του blender-http://www.blender.org

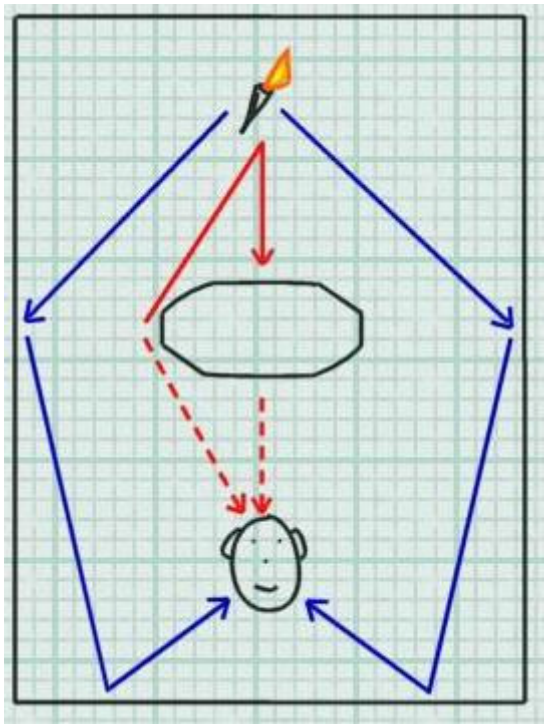
5.1 Μοντελοποίηση Του Ηχητικού Χώρου - Soundspace

Το φιλτράρισμα του περιβάλλοντος μπορεί να ενισχύσει τον ρεαλισμό ενός περιβάλλοντος με τη μίμηση των αρχιτεκτονικών χαρακτηριστικών γνωρισμάτων, όπως οι τοίχοι και οι κολόνες , που εμποδίζουν και απορροφούν τον ήχο. Οι ήχοι που είναι κρυμμένοι πίσω από μια κολόνα ή έναν τοίχο γίνονται αντιληπτοί πολύ διαφορετικά από τους ήχους που έχουν μια ανεμπόδιστη πορεία στα αυτιά του ακροατή. [58]

Υπάρχουν τρία διαφορετικά μοντέλα που αναγκάζουν μια ηχητική πηγή να γίνει αντιληπτή διαφορετικά από έναν ακροατή: Obstruction, Occlusion και Exclusion. [58]

Ηχητική παρεμπόδιση (Sound Obstruction): Έστω ότι ένα εμπόδιο τοποθετείται στη μέση ενός δωματίου, μεταξύ της ηχητικής πηγής και του ακροατή. Το απευθείας (άμεσης πορείας-πρώτο μέτωπο) του ηχητικού κύματος μπορεί να φθάσει στον ακροατή μόνο από τη μετάδοση μέσω του εμποδίου ή από την περίθλαση. Και στις δύο περιπτώσεις, μερικώς ή εντελώς θα καλυφθεί. Αυτή η επίδραση κάλυψης καλείται παρεμπόδιση. [58, 59]

Υπενθυμίζουμε ότι ένα εμπόδιο μπορεί να αναγκάσει τα ηχητικά κύματα να αλλάξουν κατεύθυνση (περίθλαση). Όσο μικρότερο είναι το μήκος κύματος (μεγαλύτερη συχνότητα), τόσο μικρότερο είναι το φαινόμενο αυτό. Το αποτέλεσμα της περίθλασης είναι ότι ο ακροατής ακούει μια άμεση ηχητική πηγή με φλιταρισμένες τις υψηλές συχνότητες. [58, 59]



Εικόνα 27 Obstruction - Όταν ένα αντικείμενο σε ένα δωμάτιο χωρίζει τον ήχο από τον ακροατή.

Οι τονικές επιδράσεις της μετάδοσης μέσω του εμποδίου ή της περίθλασης είναι παρόμοιες, επειδή τα υλικά απορροφούν περισσότερο τις υψηλές συχνότητες, απ' ότι τις χαμηλές. Επομένως και στις δύο περιπτώσεις θα χρησιμοποιηθεί low-pass filter [58, 59]

Υπάρχει μια βασική ακουστική διαφορά μεταξύ των δύο φαινομένων, ως προς την θέση της ηχητικής πηγής. Στην περίπτωση της μετάδοσης μέσω του εμποδίου ο ήχος φαίνεται ακόμα να προέρχεται ουσιαστικά από την ίδια κατεύθυνση, σαν να μην υπήρξε κανένα εμπόδιο. Στην περίπτωση της περίθλασης ο ήχος φαίνεται να έρχεται από την άκρη του εμποδίου, δημιουργώντας ένα ηχητικό είδωλο της πηγής. [58, 59]

Τα ανακλώμενα ηχητικά κύματα «κυκλοφορούν» γύρω από την ηχητική παρεμπόδιση. Ως επί το πλείστον η παρεμπόδιση εμποδίζει μόνο ένα μικροσκοπικό μέρος των ανακλάσεων και της αντήχησης της ηχητικής πηγής, με αποτέλεσμα τα ακουστικά ερεθίσματα να παραμένουν ουσιαστικά σταθερά, με ή χωρίς την παρεμπόδιση. [58]

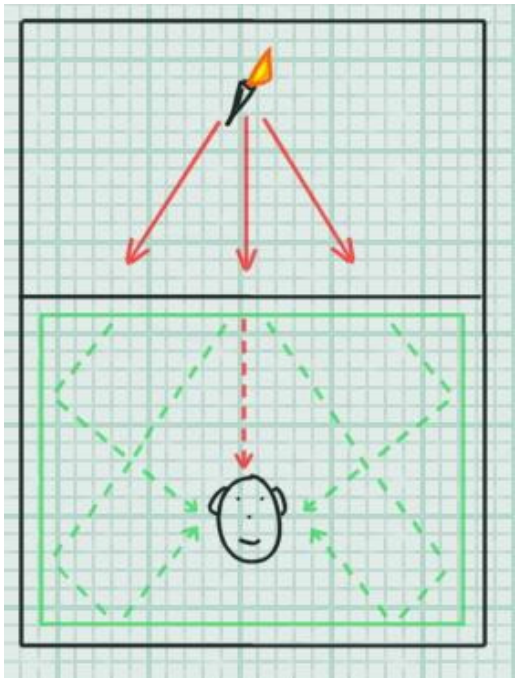
Η επίδραση της παρεμπόδισης ουσιαστικά περιορίζεται στο απευθείας μέτωπο του ήχου. Η έλλειψη (ή αλλιώς το κάλυμμα) του απευθείας ήχου, σε συνδυασμό με τις ανακλάσεις και την αντήχηση, είναι αυτό που ενημερώνει τον εγκέφαλο ότι η πηγή βρίσκεται πίσω από ένα εμπόδιο. [58]

Μη ηχητική διαπέραση (Sound Occlusion): Έστω ότι ένα τοίχος τοποθετείται στη μέση ενός δωματίου δημιουργώντας δύο δωμάτια μεταξύ της ηχητικής πηγής και του ακροατή. Σε αυτήν την περίπτωση οποιοσδήποτε ήχος που περνάει από την πηγή στον ακροατή, πρέπει να περάσει μέσω του τοίχου, ο οποίος καλύπτει τους ήχους. Διαφέρει από την παρεμπόδιση, δεδομένου ότι η παρεμπόδιση έχει ανοικτό (αν και έμμεσο) χώρο μεταξύ της πηγής και του ακροατή. [58]

Ο τοίχος που χωρίζει την ηχητική πηγή από τον ακροατή ενεργεί ως ένα μεγάλο φίλτρο. Τα άμεσα ηχητικά κύματα από την πηγή (μαζί με τη συνοδεία των ανακλάσεων και της αντήχησης) χτυπούν τον τοίχο και περνούν στην άλλη πλευρά, όπου διαχέονται στο δωμάτιο του ακροατή. [58]

Καθώς οι ήχοι περνούν μέσα από τον τοίχο, φιλτράρονται οι υψηλές συχνότητες, αφήνοντας ένα υπερβολικά «καλυμμένο» αποτέλεσμα. Το ηχητικό κύμα που περνά μέσω του τοίχου συμβάλλει και στη ανάκλαση του περιβάλλοντος του ακροατή. [58]

Όταν ο εγκέφαλος ακούει μια «καλυμμένη» ηχητική πηγή μαζί με τις «καλυμμένες» ανακλάσεις και την αντήχηση, μπορεί να αναγνωρίσει ότι η πηγή βρίσκεται πίσω από έναν τοίχο ή ένα άλλο υλικό. [58]



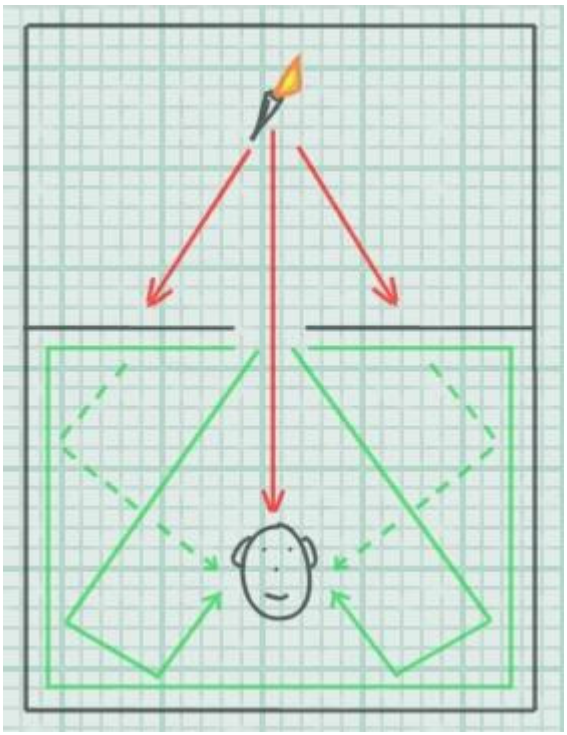
Εικόνα 28 Occlusion - Ένας πλήρης τοίχος χωρίζει την ηχητική πηγή από τον ακροατή

Αυτή είναι αντίθεση μιας μη ηχητικά διαπερατής πηγής. Ο απευθείας ήχος «καλύπτεται», αλλά δεν καλύπτονται οι ανακλάσεις και η αντήχηση, ενώ η ποιότητα της κάλυψης παρέχει στον εγκέφαλο πληροφορίες για την κατασκευή του τοίχου (εάν είναι πυκνός, λεπτός, στερεός, μαλακός, και τα λοιπά). [58]

Ηχητικός Αποκλεισμός(Sound Exclusion): Έστω ότι ένα τοίχος με ένα άνοιγμα (π.χ πόρτα) τοποθετείται στη μέση ενός δωματίου, δημιουργώντας δύο δωμάτια μεταξύ της ηχητικής πηγής και του ακροατή. Η πηγή και ο ακροατής χωρίζονται και σε αυτήν την περίπτωση από τον τοίχο, αλλά η ύπαρξη του ανοίγματος επιτρέπει στον ήχο να εισαχθεί στο δωμάτιο και, σε ορισμένες θέσεις, η απευθείας πορεία μεταξύ της πηγής και του ακροατή είναι εφικτή. [58]

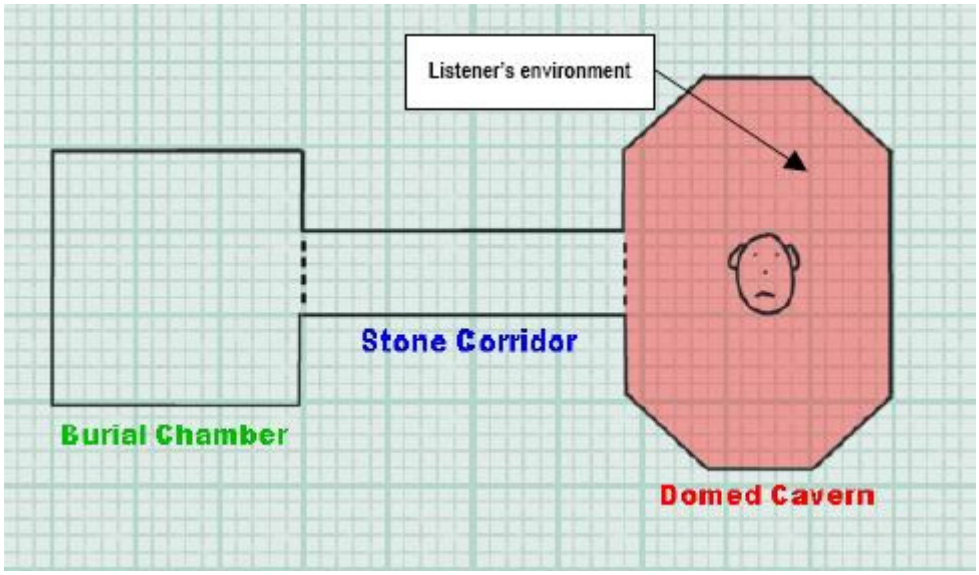
Σε αυτήν την κατάσταση, ο άμεσος ήχος δεν καλύπτεται, αλλά η πηγή μπορεί μόνο να μεταδώσει ένα μικρό ποσό ενέργειας στο δωμάτιο του ακροατή, μέσω του ανοίγματος. Το ποσοστό ανακλάσεων του ήχου που γίνεται αντιληπτό στον ακροατή από το περιβάλλον της πηγής εξαρτάται από το μέγεθος του ανοίγματος και από την απόσταση της πηγή από το άνοιγμα. [58]

Το απευθείας ηχητικό κύμα περνώντας μέσω του ανοίγματος, συμβάλλει στην αντήχηση του περιβάλλοντος του ακροατή. Η θέση του ακροατή μπορεί να είναι τέτοια, που ένα εμπόδιο να εμποδίζει την άμεση πορεία από την πηγή προς τον ακροατή (αυτό θα μπορούσε να είναι ο τοίχος που χωρίζει τα δύο δωμάτια ή κάποιο άλλο εμπόδιο που βρέθηκε στο δωμάτιο του ακροατή). Σε αυτή την περίπτωση, υπάρχει ένας συνδυασμός αποκλεισμών (που μειώνουν τον ήχο στο δωμάτιο του ακροατή) και παρεμπόδισης (που καλύπτει τη άμεση πορεία του ήχου). [58]



Εικόνα 29 Exclusion - Όταν δημιουργηθεί ένα άνοιγμα στον τοίχο που χωρίζει τον ήχο από τον ακροατή, η απευθείας πορεία, σε ορισμένες θέσεις, είναι εφικτή.

Με την χρήση ενός μόνο reverb δεν υπάρχει κανένας τρόπος να υποδειχθούν ευδιάκριτα ηχητικά πολλαπλά-περιβάλλοντα (sound Multi-Environment). [58]



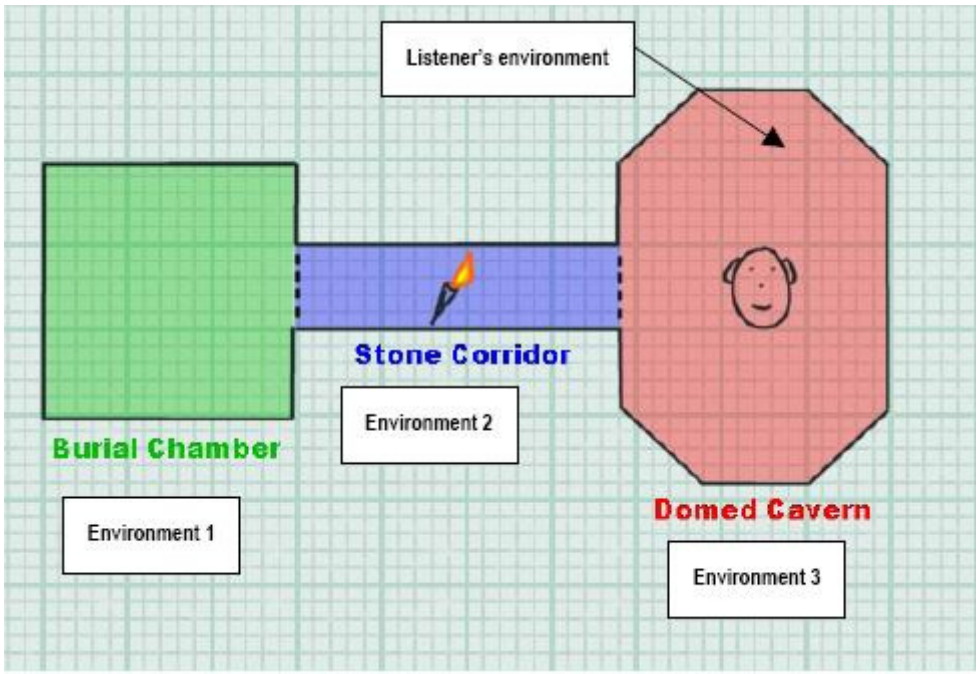
Εικόνα 30 Με ένα reverb, μόνο το περιβάλλον του ακροατή μπορεί να απεικονιστεί ηχητικά.

Εξετάζοντας την εικόνα 30 μπορούμε να παρατηρήσουμε ότι ο ακροατής έχει την δυνατότητα να ακούσει μόνο το περιβάλλον του Domed Cavern και όχι όλο το ηχητικό περιβάλλον (soundscape), δηλαδή τα αλλά δύο δωμάτια που επικοινωνούν με το Domed Cavern. [58]

Προσθέτοντας στην εικόνα τον ήχο ενός φλεγόμενου πυρσού στον διάδρομο Stone Corridor και θέλοντας να προσομοιώσουμε το ηχητικό περιβάλλον με τον ήχο του φλεγόμενου πυρσού, διαπιστώνουμε ότι καταρχήν έχουμε την απευθείας πορεία του ήχου από τον φλεγόμενο πυρσό στο αυτί του ακροατή. Επίσης ο ακροατής θα αντιληφθεί κάποια ανάκλαση ήχου με την ακουστική ιδιότητα του περιβάλλοντος της πηγής - Stone Corridor. Ταυτόχρονα οι ήχοι που παράγονται μέσα στο περιβάλλον του ακροατή - Domed Cavern-ανακλώνται από τους τοίχους παράγοντας την ακουστική επίδραση του δωματίου-Domed Cavern. Ο φλεγόμενος πυρσός θα συμβάλει και αυτός στην αντήχηση του δωματίου-Domed Cavern–, όπως όμως φτάνει στο δωμάτιο (δηλαδή τροποποιημένος από τις ανακλάσεις του δωματίου-Stone Corridor - αλλά και από τους τοίχους και τα εμπόδια που θα επέμβουν σε αυτόν). [58]

Επομένως δημιουργείται η ανάγκη δημιουργίας ενός Listener's environment, που θα αποτελείται από πολλαπλά reverbs που θα δημιουργούν πολλαπλά ηχητικά περιβάλλοντα. [58]

Με το μοντέλο πολλαπλών-περιβαλλόντων, τα περιβαλλοντικά reverbs μπορούν να δοθούν για πολλαπλές θέσεις και τα filtering effects που εφαρμόζονται να καλύψουν σωστά τον απευθείας και ανακλώμενο ήχο. [58]

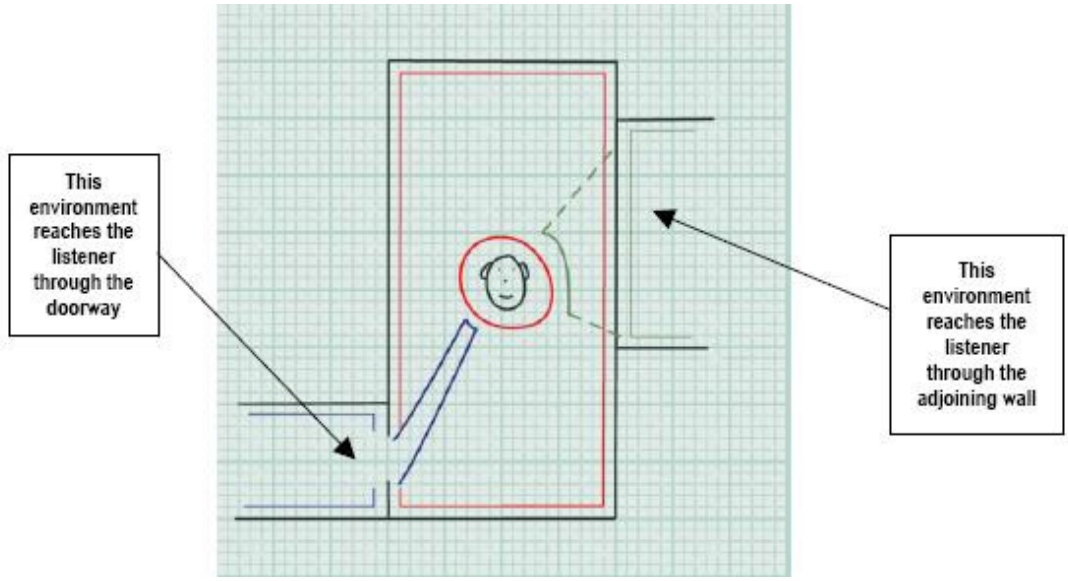


Εικόνα 31 Η χρήση πολλαπλών reverbs επιτρέπει την δημιουργία ενός ρεαλιστικού ηχητικού περιβάλλοντος.

Μια εφαρμογή που χρησιμοποιεί περισσότερα από ένα reverb μπορεί να παρέχει μια πιο ρεαλιστικότερη εμπειρία, κάνοντας δυνατή την συλλογή ακουστικών πληροφοριών από όλο το ηχητικό περιβάλλον. [58]

Εκτός από τα πολλαπλά reverbs, πρέπει η εφαρμογή να είναι σε θέση να θέσει την κατευθυντικότητα του κάθε reverb. Εάν τα αποτελέσματα από κάθε περιβαλλοντική επίδραση του reverb παρουσιαστούν αναμεμειγμένα και ομοιόμορφα γύρω από το κεφάλι του ακροατή, δεν θα υπάρχει κάποιο νόημα. Οι πρόσθετες πληροφορίες που παράγονται από την προσομοίωση των διαφορετικών ακουστικών περιβαλλόντων είναι εξαιρετικά δύσκολο να διακριθούν. Αυτό επιτυγχάνεται με panning των περιβαλλόντων. [58]

Όταν ο ακροατής ακούει τον «απεικονισμένο» ήχο από περιβάλλοντα εκτός από το δικό του, η θέση κάθε εξωτερικού περιβάλλοντος πρέπει να υποδειχθεί. Για αυτόν τον λόγο, το Effects Extension's EAX Reverb effect περιλαμβάνει τις παραμέτρους που επιτρέπουν σε μια εφαρμογή να θέσει την κατευθυντική panning των αρχικών αντανακλάσεων και τις μετέπειτα αντηχήσεις (απόκλιση-αλλοίωση). Αυτές οι παράμετροι panning ελέγχουν και την αντιληπτή κατεύθυνση και την «απόκλιση-αλλοίωση» των αντανακλάσεων και της αντήχησης. Ο έλεγχος «αποκλίσεων-αλλοιώσεων» επιτρέπει την μεταβολή από τις διάχυτες περιβάλλουσες ανακλάσεις, στις ανακλάσεις που στρέφονται σε μια επιλεγμένη κατεύθυνση. [58]



Εικόνα 32 Navigating environments.

Με τον καθορισμό αυτών των παραμέτρων σε πραγματικό χρόνο και την μεταβολή του επεξεργάσιμου ήχου, μπορούμε να υπονοήσουμε μια κατεύθυνση και μια απόσταση, που αφορούν το αντίστοιχο περιβάλλον τον ακροατή. [58]

5.2 Βασική Δομή και Περιορισμοί

Για την περιγραφή του τρισδιάστατου ηχητικού χώρου OpenAL (εκτός από τα βασικά αντικείμενα του πυρήνα AL, που ήδη έχουμε δει) εισάγονται διάφορα νέα αντικείμενα μέσω του EFX. Ουσιαστικά το EFX εισάγει νέες παραμέτρους (μέσω των νέων objects) για τα ήδη υπάρχοντα objects της OpenAL [58]

Auxiliary Effect Slot: Ένα Auxiliary Effect Slot object αντιπροσωπεύει ένα Auxiliary Slot, που περιέχει ένα effect που μπορεί να αναμιχθεί με τον ήχο από τα επιλεγμένα Sources. Ο τύπος του effect και ο ορισμός των χαρακτηριστικών που επιδρούν στην επεξεργασία, καθορίζονται από Effect Objects που τοποθετούνται στο Auxiliary Effect Slot. [58]

Effect Objects Τα Effect objects αποτελούνται από τις παραμέτρους που απαιτούνται για να καθορίσουν ένα Auxiliary Effect, δηλ. τον τύπο effect (reverb, chorus, κ.λπ.) και τις τιμές για κάθε μια από τις παραμέτρους που ελέγχουν το effect. [58]

Filter Objects:Τα filter objects περιέχουν τις πληροφορίες για να ένα filter.Δηλαδή τον τύπο του filter(low-pass, high-pass, κ.λπ) και τις τιμές για κάθε μια από τις παραμέτρους που ελέγχουν το filter(cut-off κ.λ.π). Μπορούν να χρησιμοποιηθούν για να φιλτράρουν το απευθείας ηχητικό κύμα του Source ή να χρησιμοποιηθούν για να φιλτράρουν τον ήχο που στέλνεται σε οποιοδήποτε Auxiliary Effect Slots. [58]

Το συγκεκριμένο API διαθέτει όμως έναν μεγάλο περιορισμό που δημιουργείται από το hardware του κάθε χρήστη. Διαφορετικές κάρτες ήχου μπορούν να υποστηρίξουν διαφορετικούς τύπους effects και filters, ενώ ανάλογος είναι και ο αριθμός των Auxiliary Effect Slot, που μπορούν να χρησιμοποιηθούν. [58]

Ο συγκεκριμένος περιορισμός παρέρχεται από το EAX *(Environmental Audio Extensions)* που διαθέτει η κάρτα ήχου του χρήστη. Το EAX είναι μια τεχνολογία που αναπτύσσεται αποκλειστικά από την Creative Labs. Οι καινούργιες κάρτες της Creative υποστηρίζουν EAX 5.0, ενώ πλέον θεωρείται standard για όλες τις κάρτες ήχου το EAX 2.0. [58,60,61]

Εικόνα 33 EAX - *Environmental Audio Extensions.*

Οι διαφορές στις δυνατότητες ενός συστήματος που υποστηρίζει EAX 5.0 με ένα σύστημα που υποστηρίζει EAX 2.0, είναι μεγάλες. Μια τυποποιημένη κάρτα ήχου ενός χρήστη συνήθως υποστηρίζει EAX 2.0, ένα reverb effect και ένα Low Pass filter, ενώ ταυτόχρονα υποστηρίζει την δημιουργία ενός μόνο Auxiliary Effect Slot και, προφανώς, θα μπορεί να τοποθετηθεί σε κάθε Source ένα μόνο Auxiliary Effect Slot. [58,60,61]

Είναι προφανές ότι μέσα από το συγκεκριμένο API, μόνο χρήστες που διαθέτουν το τελευταίο μοντέλο της Creative θα μπορούν να έχουν μια ρεαλιστική ηχητική προσομοίωση του ηχητικού χώρου, όπως αυτή αναπαρίσταται από το EFX . [58,60,61]

Ο πρώτος και πιο βασικός έλεγχος για υποστήριξη του EAX ξεκινάει κατά το Initializing ενός προγράμματος. Αφού ανοιχτεί ένα device, μπορεί να γίνει ερώτηση για το αν το συγκεκριμένο device υποστηρίζει την επέκταση ALC ALC_EXT_EFX [58]

```
if (alcIsExtensionPresent(pDevice, "ALC_EXT_EFX") == AL_TRUE){

      printf("EFX-EAX Extension found!\n");
}
else {

      printf("EFX-EAX Extension No found!\n");
}
```

Στην περίπτωση που η εντολή δεν επιστρέψει AL_TRUE, καμία απολύτως λειτουργία του EFX API δεν υποστηρίζεται από την κάρτα ήχου του χρήστη. Στην περίπτωση που επιστρέψει AL_TRUE ορισμένες ή όλες οι λειτουργίες της EFX υποστηρίζονται (ανάλογα με την έκδοση της EAX που υποστηρίζει η κάρτα ήχου του χρήστή).

Η OpenAL παρέχει αντίστοιχες εντολές ερώτησης και για την έκδοση του EAX που υποστηρίζει η κάρτα ήχου. [58, 35,36]

```
if (alcIsExtensionPresent ("EAX2.0") == AL_TRUE){
            printf ("\nEAX2.0.....................Yes");
}
```

Όμως η πληροφορία έκδοσης του EAX σε πολλές περιπτώσεις δεν είναι αρκετή. Γνωρίζοντας την έκδοση του EAX που διαθέτει ο χρήστης γνωρίζουμε απλά την έκδοση που αναφέρει ο κατασκευαστής ότι διαθέτει, αλλά πολλές κάρτες ήχου, ανάλογα με την κατασκευή τους, υποστηρίζουν ελλιπώς τις εκδόσεις που παρέχουν ή παρέχουν κάποιες λειτουργίες της επόμενης έκδοσης [58, 35,36, 60]

Μπορούμε να αποφύγουμε αυτήν την σύγχυση χρησιμοποιώντας το Error System της OpenAL, κάτι που συνιστάται και από το OpenAL API. Όταν το ALC_EXT_EFX υποστηρίζεται και μια λειτουργία EFX προκαλέσει ένα AL Error, τότε θεωρείται μη υποστηρίξιμη η συγκεκριμένη λειτουργία από την κάρτα ήχου. [58, 35,36, 60]

Το initialize της EFX δημιουργείται από τον προγραμματιστή. Το API παρέχει functions της μορφής : [58]

```
/* Create Effect objects. */
typedef void (__cdecl *LPALGENEFFECTS)( ALsizei n, ALuint* effects);
```

Πριν χρησιμοποιηθεί οποιαδήποτε εντολή της EFX, πρέπει να πραγματοποιηθεί η εξής διαδικασία για το κάθε object που πρόκειται να χρησιμοποιηθεί: [58]

```
// Effect objects
LPALGENEFFECTS alGenEffects = NULL;

/* Get the Effect Extension function pointers */
alGenEffects=( LPALGENEFFECTS) alGetProcAddress("alGenEffects");

/* Check function pointers are valid */
if (!(alGenEffects))
return;
```

Επομένως μετά το Initialize της AL και εφόσον επιθυμείται η χρήση της EFX, πρέπει να εκτελείται μια συνάρτηση που θα περιέχει όλες τις εντολές της EFX:[1]

```
// Effect objects
LPALGENEFFECTS alGenEffects = NULL;
LPALDELETEEFFECTS alDeleteEffects = NULL;
LPALISEFFECT alIsEffect = NULL;
LPALEFFECTI alEffecti = NULL;
LPALEFFECTIV alEffectiv = NULL;
LPALEFFECTF alEffectf = NULL;
LPALEFFECTFV alEffectfv = NULL;
LPALGETEFFECTI alGetEffecti = NULL;
LPALGETEFFECTIV alGetEffectiv = NULL;
LPALGETEFFECTF alGetEffectf = NULL;
LPALGETEFFECTFV alGetEffectfv = NULL;

//Filter objects
LPALGENFILTERS alGenFilters = NULL;
LPALDELETEFILTERS alDeleteFilters = NULL;
LPALISFILTER alIsFilter = NULL;
LPALFILTERI alFilteri = NULL;
LPALFILTERIV alFilteriv = NULL;
LPALFILTERF alFilterf = NULL;
LPALFILTERFV alFilterfv = NULL;
LPALGETFILTERI alGetFilteri = NULL;
LPALGETFILTERIV alGetFilteriv = NULL;
LPALGETFILTERF alGetFilterf = NULL;
LPALGETFILTERFV alGetFilterfv = NULL;
```

[1] Το συγκεκριμένο τμήμα κώδικα προέρχεται από το OpenAL SDK –Example EnumerateEFX

```
// Auxiliary slot object
LPALGENAUXILIARYEFFECTSLOTS alGenAuxiliaryEffectSlots = NULL;
LPALDELETEAUXILIARYEFFECTSLOTS alDeleteAuxiliaryEffectSlots = NULL;
LPALISAUXILIARYEFFECTSLOT alIsAuxiliaryEffectSlot = NULL;
LPALAUXILIARYEFFECTSLOTI alAuxiliaryEffectSloti = NULL;
LPALAUXILIARYEFFECTSLOTIV alAuxiliaryEffectSlotiv = NULL;
LPALAUXILIARYEFFECTSLOTF alAuxiliaryEffectSlotf = NULL;
LPALAUXILIARYEFFECTSLOTFV alAuxiliaryEffectSlotfv = NULL;
LPALGETAUXILIARYEFFECTSLOTI alGetAuxiliaryEffectSloti = NULL;
LPALGETAUXILIARYEFFECTSLOTIV alGetAuxiliaryEffectSlotiv = NULL;
LPALGETAUXILIARYEFFECTSLOTF alGetAuxiliaryEffectSlotf = NULL;
LPALGETAUXILIARYEFFECTSLOTFV alGetAuxiliaryEffectSlotfv = NULL;



ALboolean ALFWIsEFXSupported(){

ALCdevice *pDevice = NULL;
ALCcontext *pContext = NULL;
ALboolean bEFXSupport = AL_FALSE;


pContext = alcGetCurrentContext();

pDevice = alcGetContextsDevice(pContext);


if (alcIsExtensionPresent(pDevice, (ALCchar*)ALC_EXT_EFX_NAME)){

// Get function pointers
alGenEffects =(LPALGENEFFECTS)alGetProcAddress("alGenEffects");
alDeleteEffects=(LPALDELETEEFFECTS)alGetProcAddress("alDeleteEffects");
alIsEffect=(LPALISEFFECT )alGetProcAddress("alIsEffect");
alEffecti = (LPALEFFECTI)alGetProcAddress("alEffecti");
alEffectiv = (LPALEFFECTIV)alGetProcAddress("alEffectiv");
alEffectf = (LPALEFFECTF)alGetProcAddress("alEffectf");
alEffectfv = (LPALEFFECTFV)alGetProcAddress("alEffectfv");
alGetEffecti = (LPALGETEFFECTI)alGetProcAddress("alGetEffecti");
alGetEffectiv = (LPALGETEFFECTIV)alGetProcAddress("alGetEffectiv");
alGetEffectf = (LPALGETEFFECTF)alGetProcAddress("alGetEffectf");
alGetEffectfv = (LPALGETEFFECTFV)alGetProcAddress("alGetEffectfv");

alGenFilters = (LPALGENFILTERS)alGetProcAddress("alGenFilters");
alDeleteFilters =(LPALDELETEFILTERS)alGetProcAddress("alDeleteFilters");
alIsFilter = (LPALISFILTER)alGetProcAddress("alIsFilter");
alFilteri = (LPALFILTERI)alGetProcAddress("alFilteri");
alFilteriv = (LPALFILTERIV)alGetProcAddress("alFilteriv");
alFilterf = (LPALFILTERF)alGetProcAddress("alFilterf");
alFilterfv = (LPALFILTERFV)alGetProcAddress("alFilterfv");
alGetFilteri = (LPALGETFILTERI )alGetProcAddress("alGetFilteri");
alGetFilteriv= (LPALGETFILTERIV )alGetProcAddress("alGetFilteriv");
alGetFilterf = (LPALGETFILTERF )alGetProcAddress("alGetFilterf");
alGetFilterfv= (LPALGETFILTERFV )alGetProcAddress("alGetFilterfv");

alGenAuxiliaryEffectSlots=(LPALGENAUXILIARYEFFECTSLOTS)
                    alGetProcAddress("alGenAuxiliaryEffectSlots");
alDeleteAuxiliaryEffectSlots=(LPALDELETEAUXILIARYEFFECTSLOTS)
                    alGetProcAddress("alDeleteAuxiliaryEffectSlots");
```

```
alIsAuxiliaryEffectSlot=(LPALISAUXILIARYEFFECTSLOT)alGetProcAddress(
                              "alIsAuxiliaryEffectSlot");
alAuxiliaryEffectSloti=(LPALAUXILIARYEFFECTSLOTI)alGetProcAddress(
                               "alAuxiliaryEffectSloti");
alAuxiliaryEffectSlotiv=(LPALAUXILIARYEFFECTSLOTIV)alGetProcAddress(
                             "alAuxiliaryEffectSlotiv");
alAuxiliaryEffectSlotf=(LPALAUXILIARYEFFECTSLOTF)alGetProcAddress(
                             "alAuxiliaryEffectSlotf");
alAuxiliaryEffectSlotfv=(LPALAUXILIARYEFFECTSLOTFV)alGetProcAddress(
                             "alAuxiliaryEffectSlotfv");
alGetAuxiliaryEffectSloti=(LPALGETAUXILIARYEFFECTSLOTI)alGetProcAddress(
                             "alGetAuxiliaryEffectSloti");
alGetAuxiliaryEffectSlotiv=(LPALGETAUXILIARYEFFECTSLOTIV)
                         alGetProcAddress("alGetAuxiliaryEffectSlotiv");
alGetAuxiliaryEffectSlotf=(LPALGETAUXILIARYEFFECTSLOTF)alGetProcAddress(
                             "alGetAuxiliaryEffectSlotf");
alGetAuxiliaryEffectSlotfv=(LPALGETAUXILIARYEFFECTSLOTFV)
                         alGetProcAddress("alGetAuxiliaryEffectSlotfv");



            /* Check function pointers are valid */

            if (alGenEffects && alDeleteEffects && alIsEffect &&
                alEffecti && alEffectiv && alEffectf && alEffectfv &&
                alGetEffecti && alGetEffectiv && alGetEffectf &&
                alGetEffectfv && alGenFilters && alDeleteFilters &&
                alIsFilter && alFilteri && alFilteriv && alFilterf &&
                alFilterfv && alGetFilteri && alGetFilteriv &&
                alGetFilterf && alGetFilterfv &&alGenAuxiliaryEffectSlots
                &&alDeleteAuxiliaryEffectSlots && alIsAuxiliaryEffectSlot
                && alAuxiliaryEffectSloti && alAuxiliaryEffectSlotiv &&
                alAuxiliaryEffectSlotf && alAuxiliaryEffectSlotfv &&
                alGetAuxiliaryEffectSloti && alGetAuxiliaryEffectSlotiv
                && alGetAuxiliaryEffectSlotf &&
                alGetAuxiliaryEffectSlotfv)

                bEFXSupport = AL_TRUE;
   }


return bEFXSupport;
}
```

Ο λόγος που η συγκεκριμένη λειτουργία δεν εκτελείται εσωτερικά μέσα στο API δεν διευκρινίζεται. Πιθανότατα σε επόμενες εκδόσεις της OpenAL η συγκεκριμένη λειτουργία να εκτελείται εσωτερικά.

Εικόνα 34 EAX - *Environmental Audio Extensions.*

5.3 Effect Objects

Τα Effect objects είναι αντικείμενα που περιέχουν τις πληροφορίες των audio Effects. Ένα Effect object αποθηκεύει τον τύπο effect και έναν κατάλογο τιμών και παραμέτρων για το συγκεκριμένο effect. [58]

Τα Effect objects δημιουργούνται από την εφαρμογή χρησιμοποιώντας alGenEffects και διαγράφονται από την εφαρμογή χρησιμοποιώντας alDeleteEffects. Η λειτουργία είναι ίδια με αυτήν της δημιουργίας οποιουδήποτε OpenAL object. [58]

```
ALuint  Effect; //Effect Object
//Generate 1 Effect Object
alGenEffects(1, &Effect);
// Delete 1 Effect Object
alDeleteEffects(1, &Effect);
```

Στην συνέχεια μπορεί να γίνει έλεγχος για το αν το συγκεκριμένο όνομα αντικειμένου είναι ένα έγκυρο όνομα ενός Effect object. [58]

```
ALboolean alIsEffect( ALuint effect);
```

Μετά από την δημιουργία ενός Effect object, η εφαρμογή πρέπει να πει ποιον τύπο Effect θέλει να αποθηκεύσει στο συγκεκριμένο Effect object που δημιούργησε. Αυτό επιτυγχάνεται μέσω της ιδιότητας AL_EFFECT_TYPE. [58]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_EFFECT_TYPE	Οποιοσδήποτε από τους διαθέσιμους τύπους effect	AL_EFFECT_NULL

Οι διαθέσιμοι τύποι effects είναι :

AL_EFFECT_TYPE	ΠΕΡΙΓΡΑΦΗ
AL_EFFECT_NULL	NULL
AL_EFFECT_REVERB	Reverb
AL_EFFECT_CHORUS	Chorus
AL_EFFECT_DISTORTION	Distortion
AL_EFFECT_ECHO	Echo
AL_EFFECT_FLANGER	Flanger
AL_EFFECT_FREQUENCY_SHIFTER	Frequency Shifter
AL_EFFECT_VOCAL_MORPHER	Vocal Morpher
AL_EFFECT_PITCH_SHIFTER	Pitch Shifter
AL_EFFECT_RING_MODULATOR	Ring Modulator
AL_EFFECT_AUTOWAH	Autowah
AL_EFFECT_COMPRESSOR	Compressor
AL_EFFECT_EQUALIZER	Equalizer

Όπως έχει προαναφερθεί δεν είναι διαθέσιμοι όλοι οι τύποι Effects σε όλες τις κάρτες ήχου και επομένως κατά την τοποθέτηση ενός Effect πρέπει να γίνεται έλεγχος σφάλματος[58]

```
alEffecti(Effect, AL_EFFECT_TYPE, AL_EFFECT_REVERB);

if (alGetError() == AL_NO_ERROR){
      printf("Reverb Effect supported\n");
}
else {
      printf("Reverb Effect not supported\n");
}
```

Αν η διαδικασία αυτή επιτύχει, το συγκεκριμένο Effect Object Name (Effect) θα περιέχει ένα τύπο Effect (AL_EFFECT_REVERB) και ταυτόχρονα θα περιέχει όλες τις παραμέτρους του συγκεκριμένου τύπου Effect, στις προκαθορισμένες default τιμές τους. [58]

Για το reverb οι παράμετροι αυτοί είναι :

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_REVERB_DENSITY	[0.0, 1.0]	1.0
AL_REVERB_DIFFUSION	[0.0, 1.0]	1.0
AL_REVERB_GAIN	[0.0, 1.0]	0.32
AL_REVERB_GAINHF	[0.0, 1.0]	0.89
AL_REVERB_DECAY_TIME	[0.1, 20.0]	1.49
AL_REVERB_DECAY_HFRATIO	[0.1, 2.0]	0.83
AL_REVERB_REFLECTIONS_GAIN	[0.0, 3.16]	0.05
AL_REVERB_REFLECTIONS_DELAY	[0.0, 0.3]	0.007
AL_REVERB_LATE_REVERB_GAIN	[0.0,10.0]	1.26
AL_REVERB_LATE_REVERB_DELAY	[0.0, 0.1]	0.011
AL_REVERB_AIR_ABSORPTION_GAINHF	[0.892,1.0]	0.994
AL_REVERB_ROOM_ROLLOFF_FACTOR	[0.0. 10.0]	0.0
AL_REVERB_DECAY_HFLIMIT	AL_FALSE, AL_TRUE	AL_TRUE

Οι παράμετροι όλων των Effects, καθώς και οι default τιμές τους, παρατίθενται στο Παράρτημα 1 ΠΙΝΑΚΑΣ 3.2 EFX Define Values

Χρησιμοποιώντας την συνάρτηση των Effect που ακολουθούν τους γενικούς τύπους της AL, μπορούμε να ανασύρουμε μια τιμή ή να θέσουμε μια καινούργια τιμή σε μια παράμετρο ενός Effect. [58,35]

void al{Object}{n}{if}{v}(ALuint objectName, ALenum paramName, T values);

void alGet{Object}{n}{if}{v} (uint objectName, enum paramName, T values);

Επομένως χρησιμοποιώντας την συνάρτηση alEffectf που θέτει μια floating point παράμετρο ενός Effect object, μπορούμε να μεταβάλουμε για παράδειγμα το Decay Time του Reverb στην τιμή 20, που χαρακτηρίζει ένα μεγάλο δωμάτιο. [58]

```
alEffectf(Effect, AL_REVERB_DECAY_TIME, 20.0f);
```

Στην συνέχεια το συγκεκριμένο Effect Object μπορεί να συνδεθεί με οποιοδήποτε Auxiliary Effect Slot, όπως αναφέρεται στο υποκεφάλαιο που περιγράφει τα Auxiliary Effect Slots. [58]

5.4 Filter Objects

Τα Filter objects είναι αντικείμενα που περιέχουν τις πληροφορίες των audio Filters. Ένα Filter object αποθηκεύει τον τύπο Filter και έναν κατάλογο τιμών και παραμέτρων για το συγκεκριμένο Filter. [58]

Τα Filter objects δημιουργούνται από την εφαρμογή, χρησιμοποιώντας alGenFilters και διαγράφονται από την εφαρμογή, χρησιμοποιώντας alDeleteFilters. Η λειτουργία είναι ίδια με αυτήν της δημιουργίας οποιουδήποτε OpenAL object. [58]

```
ALuint        Filter; //Filter Object
//Generate 1 Filter Object
alGenFilters(1, &Filter);
// Delete Filter
alDeleteFilters(1, &Filter);
```

Στην συνέχεια μπορεί να γίνει έλεγχος για το αν το συγκεκριμένο όνομα αντικειμένου είναι ένα έγκυρο όνομα ενός Filter object. [58]

```
ALboolean alIsFilter( ALuint filter );
```

Μετά τη δημιουργία ένα Filter object η εφαρμογή πρέπει να πει ποιον τύπο Filter θέλει να αποθηκεύσει στο συγκεκριμένο Filter object που δημιούργησε. Αυτό επιτυγχάνεται μέσω της ιδιότητας AL_FILTER_TYPE. [58]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_FILTER_TYPE	Οποιοσδήποτε από τους διαθέσιμους τύπους Filter	AL_FILTER_NULL

Οι διαθέσιμοι τύποι Filters είναι :

AL_EFFECT_TYPE	ΠΕΡΙΓΡΑΦΗ
AL_FILTER_NULL	NULL
AL_FILTER_LOWPASS	Low Pass
AL_FILTER_HIGHPASS	High Pass
AL_FILTER_BANDPASS	Band Pass

Όπως έχει προαναφερθεί δεν είναι διαθέσιμοι όλοι οι τύποι Filter σε όλες τις κάρτες ήχου και επομένως πρέπει κατά την τοποθέτηση ενός Filter να γίνεται έλεγχος σφάλματος[58]

```
alFilteri(Filter, AL_FILTER_TYPE, AL_FILTER_LOWPASS);

if (alGetError() == AL_NO_ERROR){
      printf("\n Low Pass Filter supported ");
}
else{
      printf("\n Low Pass Filter not supported ");
}
```

Αν η διαδικασία αυτή επιτύχει, το συγκεκριμένο Filter Object Name (Filter) θα περιέχει ένα τύπο Filter (`AL_FILTER_LOWPASS`) και ταυτόχρονα θα περιέχει όλες τις παραμέτρους του συγκεκριμένου τύπου Filter στις προκαθορισμένες default τιμές τους. [58]

Οι παράμετροι αυτοί είναι :

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
Lowpass parameters		
AL_LOWPASS_GAIN	[0.0, 1.0]	1.0
AL_LOWPASS_GAINHF	[0.0, 1.0]	1.0
Highpass Parameters		
AL_HIGHPASS_GAIN	[0.0, 1.0]	1.0
AL_HIGHPASS_GAINLF	[0.0, 1.0]	1.0
Bandpass Parameters		
AL_BANDPASS_GAIN	[0.0, 1.0]	1.0
AL_BANDPASS_GAINLF	[0.0, 1.0]	1.0
AL_BANDPASS_GAINHF	[0.0, 1.0]	1.0

Πρέπει να σημειωθεί ότι στην έκδοση 1.1 της OpenAL, αν και παρέχονται οι τιμές για τα High Pass και Band Pass Filter, δεν είναι ακόμα διαθέσιμα. Πολλά σημεία του συγκεκριμένου API είναι σε ανάπτυξη αυτήν την περίοδο και αναμένονται να λειτουργήσουν σε επόμενες εκδόσεις. [58]

Επομένως αν θέλαμε να μεταβάλουμε τις τιμές του συγκεκριμένου Filter Object Name (Filter), θα χρησιμοποιούσαμε τον ακόλουθο κώδικα :

```
alFilterf(Filter, AL_LOWPASS_GAIN, 0.5f);
alFilterf(Filter, AL_LOWPASS_GAINHF, 0.5f);
```

Η αλλαγή των τιμών μιας παραμέτρου ενός Filter Object, αφότου έχει συνδεθεί με ένα Source, δεν έχει επιπτώσεις στο Source. Για την ενημέρωση του Filter που χρησιμοποιείται από ένα Source μια εφαρμογή πρέπει να ενημερώσει τις παραμέτρους ενός Filter Object και έπειτα να το επανασυνδέσει στο Source. [58]

Τα Filter objects μπορούν να χρησιμοποιηθούν με δύο διαφορετικούς τρόπους, να φιλτράρουν ένα Source ή να φιλτράρουν αυτό που στέλνεται από το Source σε ένα Auxiliary Effect Slot. [58]

Όταν ένα Filter object είναι συνδεμένο με ένα Source ως Direct Filter (με την ιδιότητα AL_DIRECT_FILTER), το φιλτράρισμα εφαρμόζεται στο άμεσο (dry) σήμα του συγκεκριμένου Source μόνο. Είναι προφανές ότι η συγκεκριμένη λειτουργία πραγματοποιείται αφού ένας buffer έχει συνδεθεί στο συγκεκριμένο Source. [58]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_DIRECT_FILTER	Filter ID	AL_FILTER_NULL

```
//To attach Filter 'Filter' to Source 'Source':
alSourcei(Source, AL_DIRECT_FILTER, Filter);
//To remove a Filter from Source 'Source':
alSourcei(Source, AL_DIRECT_FILTER, AL_FILTER_NULL);
```

Όταν ένα Filter object συνδέεται με ένα Source σαν Auxiliary Send Filter (με την ιδιότητα AL_AUXILIARY_SEND_FILTER), το φιλτράρισμα εφαρμόζεται στο σήμα που στέλνεται στο Auxiliary Effect Slot, όπως αναφέρεται στο υποκεφάλαιο των Auxiliary Effect Slot. [58]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_AUXILIARY_SEND_FILTER	Filter ID	AL_FILTER_NULL

```
alSource3i(Source,AL_AUXILIARY_SEND_FILTER, EffectSlot[0],0, Filter);
```

5.5 Auxiliary Effect Slot

Τα Auxiliary Effect Slots είναι ουσιαστικά Auxiliary Slots για τα DSP (Digital Signal Processing) Effects. Τοποθετούνται στο τέλος της επεξεργασίας σήματος που σημαίνει ότι η output μιας Auxiliary Effect Slot στέλνεται κατευθείαν στο τελικό output mix. [58]

Τα Auxiliary Effect Slots δημιουργούνται από την εφαρμογή χρησιμοποιώντας alGenAuxiliaryEffectSlots, και διαγράφονται από την εφαρμογή χρησιμοποιώντας alDeleteAuxiliaryEffectSlots. Η λειτουργία είναι ίδια με αυτήν της δημιουργίας οποιουδήποτε OpenAL object. [58]

```
ALuint EffectSlot;

//Generated Auxiliary Effect Slots
alGenAuxiliaryEffectSlots(1, &EffectSlot);
//Delete Auxiliary Effect Slots
alDeleteAuxiliaryEffectSlots(1, EffectSlot);
```

Η EFX δεν επιβάλλει οποιαδήποτε όρια στον αριθμό Auxiliary Effect Slots που μπορούν να δημιουργηθούν. Μια εφαρμογή όμως πρέπει να αναμείνει ότι οι διαφορετικές κάρτες ήχου θα υποστηρίζουν διαφορετικό αριθμό Auxiliary Effects Slots. Ο αριθμός των Auxiliary Effect Slots που υποστηρίζει μια κάρτα ήχου εντοπίζεται όταν ζητηθεί από την εφαρμογή να δημιουργεί ένα πραγματικά μεγάλο αριθμό Auxiliary Effect Slots, όπως φαίνεται στον κώδικα που ακολουθεί: [58]

```
ALuint      EffectSlot[120] = { 0 };//Test Auxiliary Effect Slots
ALuint      Loop;

//Try to create 120 Auxiliary Effect Slots
alGetError();//Clear AL error code
for (Loop = 0; Loop < 120; Loop++){
      alGenAuxiliaryEffectSlots(1, &EffectSlot[Loop]);
      if (alGetError() != AL_NO_ERROR)
            break;
}
printf("\n\n Device supports %d Auxiliary Effect Slots\n", Loop);
```

Πολλαπλά Sources μπορούν να στείλουν στο ίδιο Auxiliary Effect Slot, ωστόσο υπάρχουν περιορισμοί στον αριθμό των Auxiliary Effect Slots που μπορεί να χρησιμοποιήσει ταυτόχρονα ένα Source. Ο αριθμός των Auxiliary Effect Slots που μπορεί να χρησιμοποιήσει ένα Source μπορεί να εντοπιστεί χρησιμοποιώντας την ALC_MAX_AUXILIARY_SENDS: [58]

```
ALint Sends;//Auxiliary Send per Source

//Test max auxiliary send per source
alcGetIntegerv(pDevice, ALC_MAX_AUXILIARY_SENDS, 1, &Sends);
printf("Device supports %d Auxiliary Sends per Source\n", Sends);
```

Τα Effects φορτώνονται στα Auxiliary Effect Slots συνδέοντας τα Effect Objects στα Auxiliary Effect Slots. Μέσω της συνάρτησης alAuxiliaryEffectSloti η συνάρτηση αυτή ακολουθεί την γενική μορφή των συναρτήσεων της AL[58,35]

```
void al{Object}{n}{if}{v}(ALuint objectName, ALenum paramName, T values );
```

```
ALvoid alAuxiliaryEffectSloti( ALuint auxiliaryeffectslotName, ALenum param,
                               ALint iValue);
```

Επομένως, όταν θέλουμε να συνδέσουμε ένα Effect Object σε ένα Auxiliary Effect Slot, μπορούμε να χρησιμοποιήσουμε την παρακάτω γραμμή κώδικα : [58]

```
alAuxiliaryEffectSloti(EffectSlot[0], AL_EFFECTSLOT_EFFECT, Effect)

if (alGetError() == AL_NO_ERROR)
      printf("Successfully loaded effect into effect slot\n");
```

Το Effect Object αποθηκεύει τον τύπο του effect καθώς και τις τιμές για όλες τις παραμέτρους του συγκεκριμένου effect. Όταν ένα Effect Object συνδέεται με ένα Auxiliary Effect Slot, ένας έλεγχος πραγματοποιείται για το αν ο συγκεκριμένος τύπος effect έχει ήδη φορτωθεί στον Auxiliary Effect Slot. Αν όχι, το συγκεκριμένο effect φορτώνεται στο Auxiliary Effect Slot. [58]

Μόλις συνδεθεί το effect στο Auxiliary Effect Slot, οι παράμετροι για το effect (που αποθηκεύεται στο Effect Object) θα εφαρμοστούν στο πραγματικό device effect της κάρτας ήχου. Κάθε εφαρμογή πρέπει να ελέγξει αν η λειτουργία σύνδεσης είναι επιτυχής. Επειδή ορισμένες κάρτες ήχου μπορεί να διαθέτουν περιορισμούς στους πόρους τους, αυτό σημαίνει ότι μπορεί να μην μπορούν όλοι οι τύποι Effect να φορτωθούν σε Auxiliary Effect Slots. [58]

Η αλλαγή μιας παραμέτρου στο Effect Object, αφότου έχει συνδεθεί με το Auxiliary Effect Slot, δεν έχει επίδραση στο Auxiliary Effect Slot. Για την ενημέρωση τέτοιων αλλαγών μια εφαρμογή πρέπει να ενημερώσει τις παραμέτρους ενός Effect Object και έπειτα να επανασυνδέσει το Effect Object στο Auxiliary Effect Slot. [58]

Για να ελέγξουμε αν ένα συγκεκριμένο όνομα Auxiliary Effect Slot είναι ένα έγκυρο Auxiliary Effect Slot, εκτελούμε την παρακάτω εντολή:

```
ALboolean alIsAuxiliaryEffectSlot( ALuint auxiliaryeffectslotName);
```

Για να αποσυνδεθεί ένα Effect από μια Auxiliary Effect Slot, η εφαρμογή μπορεί να συνδέσει το κενό Effect object

```
alAuxiliaryEffectSloti(EffectSlot[0],AL_EFFECTSLOT_EFFECT,AL_EFFECT_NULL)
```

Μια εφαρμογή μπορεί να ελέγξει το output gain μιας Auxiliary Effect Slot. Η τιμή μηδέν (0) ισοδυναμεί σε κλείσιμο του Auxiliary Effect Slot. [58]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_EFFECTSLOT_GAIN	0.0 έως 1.0	1.0

```
/* Set Aux Effect Slot Gain to 0.5*/
alAuxiliaryEffectSlotf(EffectSlot[0], AL_EFFECTSLOT_GAIN, 0.5f);
```

Επίσης είναι δυνατόν να ρυθμιστεί αν το Auxiliary Effect Slot Object θα στέλνει αυτόματες ρυθμίσεις βασισμένες στις φυσικές θέσεις των sources και του listener. Αυτή η ιδιότητα πρέπει να επιτραπεί όταν μια εφαρμογή επιθυμεί να χρησιμοποιήσει ένα reverb effect για να μιμηθεί το περιβάλλον που περιβάλλει έναν listener ή ένα σύνολο από sources. [58]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_EFFECTSLOT_AUXILIARY_SEND_AUTO	AL_TRUE AL_FALSE	AL_TRUE

```
/* Set Aux Effect Slot Send Auto flag to true */
alAuxiliaryEffectSloti(EffectSlot[0],AL_EFFECTSLOT_AUXILIARY_SEND_AUTO,AL_TRUE);
```

Ένας Source μπορεί να ρυθμιστεί έτσι ώστε να στέλνει το περιεχόμενό του σε Auxiliary Effect Slot Object. Η ρύθμιση αυτή επιτυγχάνεται μέσω της ιδιότητας AL_AUXILIARY_SEND_FILTER. Η συγκεκριμένη ιδιότητα χρησιμοποιείται και για την αποστολή filter στο Auxiliary Effect Slot Object. [58]

ΟΝΟΜΑ	ΤΙΜΕΣ	DEFAULT
AL_AUXILIARY_SEND_FILTER	Filter ID	AL_FILTER_NULL

Είναι προφανές ότι πριν από την συγκεκριμένη λειτουργία πρέπει να έχει γίνει ερώτηση για τον αριθμό των Auxiliary Effect Slots, που μπορεί να χρησιμοποιήσει ταυτόχρονα ένα Source, όπως αναφέρεται παραπάνω. [58]

Για να διαμορφώσουμε έναν Source να στείλει σε ένα Auxiliary Effect Slot χωρίς φιλτράρισμα (χρησιμοποιώντας auxiliary send 0), χρησιμοποιούμε την παρακάτω εντολή: [58]

```
alSource3i(Source,AL_AUXILIARY_SEND_FILTER, EffectSlot[0],0,AL_FILTER_NULL);
```

Για να διαμορφώσουμε έναν Source να στείλει σε ένα Auxiliary Effect Slot με φιλτράρισμα (χρησιμοποιώντας auxiliary send 0) χρησιμοποιούμε την παρακάτω εντολή: [58]

```
alSource3i(Source,AL_AUXILIARY_SEND_FILTER, EffectSlot[0],0, Filter);
```

Σε αυτήν την περίπτωση, στο Auxiliary Effect Slot Object εφαρμόζεται και το περιεχόμενο του Source, αλλά και το περιεχόμενο ενός Filter Object (όπως αναλύεται στο κεφαλαίο των Filter Object). [58]

Για να αποσυνδέσουμε ένα Auxiliary Effect Slot από έναν Source, στέλνουμε τις τιμές NULL, όπως φαίνεται παρακάτω: [58]

```
alSource3i(Source,AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,AL_FILTER_NULL);
```

Ένας πλήρης κατάλογος των λειτουργιών της EFX παρουσιάζεται στο Παράρτημα 1:

ΠΙΝΑΚΑΣ 3.1: EFX Functions
ΠΙΝΑΚΑΣ 3.2 EFX Define Values

6. ALUT API: The OpenAL Utility Toolkit

Το ALUT API αποτελεί ένα Utility Toolkit για την OpenAL (αντιστοιχεί στο GLUT API για την OpenGL). Η τελευταία έκδοση είναι η 1.1.0, όπου και διαχωρίστηκε από την OpenAL και πλέον παρέχεται ως ξεχωριστή βιβλιοθήκη. [45]

Το συγκεκριμένο API είναι ένα βοηθητικό API που παρέχει εντολές διευκολύνσεων στους νέους προγραμματιστές. Σκοπός του είναι να εισαγάγει τους προγραμματιστές στο περιβάλλον της OpenAL, δίνοντάς τους την δυνατότητα να χειριστούν τον πυρήνα AL, χωρίς να χρησιμοποιήσουν καθόλου τις εντολές της ALC. Ταυτόχρονα παρέχει και ιδιαίτερα εύχρηστα εργαλεία που μπορούν να χρησιμοποιούνται και από προγραμματιστές, που ήδη γνωρίζουν πλήρως το OpenAL API.[45]

Η πρώτη διευκόλυνση που παρέχει το ALUT API, φαίνεται στο Initialization και Exit, παρέχοντας τρεις εντολές για την συγκεκριμένη διαδικασία: [45]

- Στην περίπτωση που ο προγραμματιστής επιθυμεί το Initialization χωρίς να χρησιμοποιήσει καθόλου την ALC, μπορεί να εκτελέσει τον ακόλουθο κώδικα:

```
alutInit (NULL, NULL)
```

  Η παραπάνω εντολή αρχικοποιεί τις τιμές για την ALUT και δημιουργεί ένα τρέχον OpenAL context για το default device. [45]

- Στην περίπτωση που ο προγραμματιστής επιθυμεί να χρησιμοποιήσει την ALUT, αλλά θέλει το Initialization να γίνει μέσα από την ALC, μπορεί να εκτελέσει τον ακόλουθο κώδικα, αφού έχει ολοκληρώσει το Initialization μέσω της ALC:

```
alutInitWithoutContext (NULL,NULL);
```

  Η παραπάνω εντολή αρχικοποιεί τις τιμές για την ALUT, αλλά δεν δημιουργεί κάποιο τρέχον OpenAL context για το default device. Έτσι αυτό πρέπει να γίνει μέσω των συνηθισμένων ALC κλήσεων. [45]

- Στην περίπτωση που ο προγραμματιστής επιθυμεί το Exit, μπορεί να εκτελέσει τον ακόλουθο κώδικα:

```
alutExit()
```

  Η παραπάνω εντολή κλείνει οποιαδήποτε OpenAL device / context έχουν δημιουργηθεί με την alutIni, αλλά όχι αυτά που η εφαρμογή δημιούργησε με την χρήση της ALC (επομένως αυτά πρέπει να κλείσουν χρησιμοποιώντας την ALC με τους τρόπους που έχουν αναφερθεί). [45]

Το παρακάτω τμήμα κώδικα εκτελεί την λειτουργία των Initialization και Exit χωρίς την χρήση της βιβλιοθήκης ALC.

```
int main ()
{
ALenum Error; /*Save ALUT error status*/

  printf ( "Initialization Alut..............." );
  alutGetError();/*Clear error code*/
  if (!alutInit (NULL ,NULL)){
        Error = alutGetError();
        printf("Error: %s\n", alutGetErrorString(Error));
        return 0;
    }

  /* ...Processing Loop ... */

  printf ( "Exit Alut........................." );
  alutGetError();/*Clear error code*/
  if (!alutExit ())
    {
        Error = alutGetError();
        printf("Error: %s\n", alutGetErrorString(Error));
        return 0;
    }

  return 0;
}
```

Παραδείγματα κώδικα που χρησιμοποιούν την alutInitWithoutContext σε συνδυασμό με την ALC, παρουσιάζονται στο ΚΕΦΑΛΑΙΟ 3 από το παράδειγμα 3. Example: Simple Static Sound και μετά.

Το πιο βασικό εργαλείο που παρέχει η ALUT είναι η λειτουργία του Load Data. Η πιο απλή μορφή αυτής της διαδικασίας φαίνεται στον παρακάτω κώδικα. Η εφαρμογή δημιουργεί έναν buffer από ένα έτοιμο Hello world αρχείο, που παρέχεται μέσα από την εντολή alutCreateBufferHelloWorld.[45]

```
ALuint helloBuffer, helloSource;
printf ( "Initialization Alut..............\n" );
alutInit (NULL, NULL);

printf ( "Play Hello world..............\n" );
helloBuffer = alutCreateBufferHelloWorld ();
alGenSources (1, &helloSource);
alSourcei (helloSource, AL_BUFFER, helloBuffer);
alSourcePlay (helloSource);

/*Σταματάει την εφαρμογή για έναν δεδομένο αριθμό δευτερολέπτων*/
alutSleep (1);
printf ( "Exit Alut........................\n" );
alutExit ();
printf ( "\n\n\n\n\n Press [Enter] to quit . . ." );
fflush ( stdout );
getchar();
return 0;
```

Η ALUT επιτρέπει την δημιουργία buffer κυματομορφών μέσω της εντολής: [45]

```
ALuint alutCreateBufferWaveform (ALenum waveshape,
                                 ALfloat frequency,
                                 ALfloat phase,
                                 ALfloat duration);
```

Όπου:

- *Waveshape* είναι ένας από τους ακόλουθους τύπους κυματομορφών: `ALUT_WAVEFORM_SINE` , `ALUT_WAVEFORM_SQUARE`, `ALUT_WAVEFORM_SAWTOOTH`, `ALUT_WAVEFORM_WHITENOISE`, `ALUT_WAVEFORM_IMPULSE`
- *Frequency* είναι η επιθυμητή συχνότητα σε Hertz
- *Phase* είναι η φάση της κυματομορφής από -180 έως +180
- *duration* είναι η διάρκεια σε δευτερόλεπτα

Πρέπει να σημειωθεί ότι η συχνότητα και η φάση αγνοούνται για κυματομορφές λευκού θορύβου. [45]

Αντίστοιχα, μπορούμε να δημιουργήσουμε ένα buffer από ένα αρχείο ήχου, μέσω της εντολής: [45]

```
ALuint alutCreateBufferFromFile (const char *filename);
```

Προ στο παρόν η ALUT έχει την δυνατότητα να φορτώσει wav αρχεία ήχου. Άλλοι τύποι αρχείων μπορούν να χρησιμοποιηθούν με την βοήθεια audio encoder API, όπως αρχεία τύπου ogg με την βοήθεια του API Ogg Vorbis, που παρέχεται από την διεύθυνση http://www.vorbis.com/ [45]

Ένας πλήρης κατάλογος των λειτουργιών της ALUT παρουσιάζεται στο Παράρτημα 1:

ΠΙΝΑΚΑΣ 4.1: ALUT Functions list
ΠΙΝΑΚΑΣ 4.2 ALUT Define Values

ΚΕΦΑΛΑΙΟ 3
ΟΛΟΚΛΗΡΩΜΕΝΕΣ ΥΛΟΠΟΙΗΣΕΙΣ ΣΤΟ ΠΕΡΙΒΑΛΛΟΝ ΤΗΣ OPENAL
Creative Labs Inc. OpenAL Version 1.1

1. Example: Router test

Το παράδειγμα αυτό προέρχεται από το CVS (Control Version System http://www.nongnu.org/cvs/) της OpenAL και αποτελεί μια εφαρμογή δοκιμής: Router Test. Το OpenAL checkout παρέχεται στην διεύθυνση: http://www.openal.org/repos/openal/. Είναι το μέρος που ο καθένας μπορεί να βρει και να δει το πως αναπροσαρμόζεται η OpenAL σε επίπεδο development source code για την επόμενη έκδοσή της.

Από το συγκεκριμένο παράδειγμα έχουν αφαιρεθεί (παρουσιάζονται ως σχόλια) τα τμήματα του κώδικα της OpenAL που δεν λειτουργούν στην επίσημη έκδοσή της, δηλαδή το *TEST : Enumerate *all* the playback devices* και *TEST : Get Default *All* Playback Device* που περιέχουν τις εντολές ALC_ENUMERATE_ALL_EXT και ALC_DEFAULT_ALL_DEVICES_SPECIFIER, που, προφανώς, θα παρέχονται στην επόμενη επίσημη έκδοση της OpenAL.

Σε αυτή τη δομή του κώδικα αναλύονται πλήρως οι λειτουργιες των devices και το πώς η Openal τα διαχειρίζεται, ενώ επίσης παρουσιάζονται όλες οι δυνατές υλοποιήσεις των συναρτήσεων *alcGetString* και *alcIsExtensionPresent.* Επίσης χρησιμοποιούνται οι εντολές *waveOutGetNumDevs* και *waveInGetNumDevs,* που δεν παρέχονται από την OpenAL, αλλά από το *winmm.lib,* το οποίο παρέχεται μαζί με το Microsoft Platform SDK.

```
#include <al.h>
#include <alc.h>
#include <stdio.h>
#include <windows.h>
#include <mmsystem.h>

int main(int argc, char* argv[])
{
const ALchar *szNames = NULL;
long lErrorCount = 0;
```

```
//////////////////////////////////////////////////////////////////////
// TEST: Enumerate the playback devices
printf("-------------------------------------\n");
printf("TESTING ALC_ENUMERATION_EXT EXTENSION\n\n");
if (alcIsExtensionPresent(NULL, "ALC_ENUMERATION_EXT") == AL_TRUE)
{
	printf("ALC_ENUMERATION_EXT Device List:-\n\n");

	szNames = alcGetString(NULL, ALC_DEVICE_SPECIFIER);
	if (strlen(szNames) == 0)
		printf("NO DEVICES FOUND\n");
	else
	{
		while (szNames && *szNames)
		{
			printf("%s\n", szNames);
			szNames += (strlen(szNames) + 1);
		}
	}
}
else
{	printf("!!!ERROR!!! : ALC_ENUMERATION_EXT NOT FOUND!\n");
	lErrorCount++;
}
printf("-------------------------------------\n\n");
```

```
////////////////////////////////////////////////////////////////////
// TEST : Get Default Playback Device
//

printf("--------------------------------------\n");
printf("TESTING GET DEFAULT PLAYBACK DEVICE\n\n");
szNames = alcGetString(NULL, ALC_DEFAULT_DEVICE_SPECIFIER);
if (szNames && strlen(szNames))
{
	printf("\n DEFAULT DEVICE is %s\n", szNames);
}
else
{
	if (waveOutGetNumDevs())
	{
		printf("\n !!!ERROR!!! DEFAULT DEVICE NOT FOUND!\n");
		lErrorCount++;
	}
	else
	{
		printf("\n DEFAULT DEVICE NOT FOUND!\n");
	}
}
printf("--------------------------------------\n\n");

////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
// TEST : Enumerate all the capture devices
//

printf("--------------------------------------\n");
printf("TESTING CAPTURE ENUMERATION EXTENSION\n\n");
if (alcIsExtensionPresent(NULL, "ALC_ENUMERATION_EXT") == AL_TRUE)
{
	printf("ALC_ENUMERATION_EXT Capture Device List:-\n\n");

	szNames = alcGetString(NULL, ALC_CAPTURE_DEVICE_SPECIFIER);
	if (strlen(szNames) == 0)
		printf("NO DEVICES FOUND\n");
	else
	{
		while (szNames && *szNames)
		{
			printf("%s\n", szNames);
			szNames += (strlen(szNames) + 1);
		}
	}
}
else
{
	printf("!!!ERROR!!! : ALC_ENUMERATION_EXT NOT FOUND!\n");
	lErrorCount++;
}
printf("--------------------------------------\n\n");
////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
// TEST : Get Default Capture Device
//
printf("----------------------------------------\n");
printf("TESTING DEFAULT CAPTURE DEVICE\n\n");
szNames = alcGetString(NULL, ALC_CAPTURE_DEFAULT_DEVICE_SPECIFIER);
if (szNames && strlen(szNames))
{
	printf("\nDEFAULT CAPTURE DEVICE IS %s\n", szNames);
}
else
{
	if (waveInGetNumDevs())
	{
		printf("\n!!!ERROR!!! DEFAULT CAPTURE DEVICE NOT FOUND!\n");
		lErrorCount++;
	}
	else
	{
		printf("\nDEFAULT CAPTURE DEVICE NOT FOUND!\n");
	}
}
printf("----------------------------------------\n\n");
////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
// TEST : Enumerate *all* the playback devices
//
//*ALC_ALL_DEVICES_SPECIFIER not exist in ALC OpenAL 1.1
//ONLY WORK WITH SVN ALC.H

printf("----------------------------------------\n");
printf("TESTING PLAYBACK ENUMERATE ALL EXTENSION\n\n");
if (alcIsExtensionPresent(NULL, "ALC_ENUMERATE_ALL_EXT") == AL_TRUE)
{
	printf("ALC_ENUMERATE_ALL_EXT DEVICE LIST:-\n\n");

	szNames = alcGetString(NULL, ALC_ALL_DEVICES_SPECIFIER);
	if (strlen(szNames) == 0)
		printf("NO DEVICES FOUND\n");
	else
	{
		while (szNames && *szNames)
		{
			printf("%s\n", szNames);
			szNames += (strlen(szNames) + 1);
		}
	}
}
else
{
	printf("!!!ERROR!!! : ALC_ENUMERATE_ALL_EXT NOT FOUND!\n");
	lErrorCount++;
}
printf("----------------------------------------\n\n");
*/

////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
// TEST : Get Default *All* Playback Device
//
//*ALC_DEFAULT_ALL_DEVICES_SPECIFIER not exist in ALC OpenAL 1.1
//ONLY WORK WITH SVN ALC.H

printf("----------------------------------------\n");
printf("TESTING DEFAULT ALL PLAYBACK DEVICE\n\n");
szNames = alcGetString(NULL, ALC_DEFAULT_ALL_DEVICES_SPECIFIER);
if (szNames && strlen(szNames))
{
	printf("\n DEFAULT ALL DEVICES IS %s\n", szNames);
	}
	else
	{
	if (waveOutGetNumDevs())
	{
		printf("\n !!!ERROR!!! DEFAULT ALL DEVICE NOT FOUND!\n");
		lErrorCount++;
	}
	else
	{
		printf("\nDEFAULT ALL DEVICES NOT FOUND!\n");
	}
}
printf("---------------------------------------\n\n");
*/

////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
// TEST : Open 'Generic Hardware' device
//

printf("----------------------------------------\n");
printf("TESTING 'Generic Hardware' DEVICE\n\n");
ALCdevice *pDevice = alcOpenDevice("Generic Hardware");
if (pDevice)
{
	printf("OPENED 'Generic Hardware' DEVICE ... GOT %s\n",
alcGetString(pDevice, ALC_DEVICE_SPECIFIER));
	alcCloseDevice(pDevice);
}
else
{
	if (waveOutGetNumDevs())
	{
		printf("!!!ERROR!!! : FAILED TO OPEN 'Generic Hardware'
DEVICE\n");
		lErrorCount++;
	}
	else
	{
		printf("FAILED TO OPEN 'Generic Hardware' DEVICE\n");
	}
}
printf("---------------------------------------\n\n");

////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
// TEST : Open 'Generic Software' device
//

printf("-------------------------------------\n");
printf("TESTING 'Generic Software' DEVICE\n\n");
pDevice = alcOpenDevice("Generic Software");
if (pDevice)
{
	printf("OPENED 'Generic Software' DEVICE ... GOT %s\n",
alcGetString(pDevice, ALC_DEVICE_SPECIFIER));
	alcCloseDevice(pDevice);
}
else
{
	if (waveOutGetNumDevs())
	{
		printf("!!!ERROR!!! : FAILED TO OPEN 'Generic Software'
DEVICE\n");
		lErrorCount++;
	}
	else
	{
		printf("FAILED TO OPEN 'Generic Software' DEVICE\n");
	}
}
printf("-------------------------------------\n\n");

////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
// TEST : Open legacy 'DirectSound3D' device
//

printf("-------------------------------------\n");
printf("TESTING LEGACY 'DirectSound3D' DEVICE\n\n");
pDevice = alcOpenDevice("DirectSound3D");
if (pDevice)
{
	printf("OPENED 'DirectSound3D' DEVICE ... GOT %s\n",
alcGetString(pDevice, ALC_DEVICE_SPECIFIER));
	alcCloseDevice(pDevice);
}
else
{
	if (waveOutGetNumDevs())
	{
		printf("!!!ERROR!!! : FAILED TO OPEN 'DirectSound3D'
DEVICE\n");
		lErrorCount++;
	}
	else
	{
		printf("FAILED TO OPEN 'DirectSound3D' DEVICE\n");
	}
}
printf("-------------------------------------\n\n");

////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
// TEST : Open legacy 'DirectSound' device
//

printf("-------------------------------------\n");
printf("TESTING LEGACY 'DirectSound' DEVICE\n\n");
pDevice = alcOpenDevice("DirectSound");
if (pDevice)
{
	printf("OPENED 'DirectSound' DEVICE ... GOT %s\n",
alcGetString(pDevice, ALC_DEVICE_SPECIFIER));
	alcCloseDevice(pDevice);
}
else
{
	if (waveOutGetNumDevs())
	{
		printf("!!!ERROR!!! : FAILED TO OPEN 'DirectSound' DEVICE\n");
		lErrorCount++;
	}
	else
	{
		printf("FAILED TO OPEN 'DirectSound' DEVICE\n");
	}
}
printf("-------------------------------------\n\n");

////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
// TEST: Open legacy 'MMSYSTEM' device
//

printf("-------------------------------------\n");
printf("TESTING LEGACY 'MMSYSTEM' DEVICE\n\n");
pDevice = alcOpenDevice("MMSYSTEM");
if (pDevice)
{
	printf("OPENED 'MMSYSTEM' DEVICE ... GOT %s\n", alcGetString(pDevice,
ALC_DEVICE_SPECIFIER));
	alcCloseDevice(pDevice);
}
else
{
	if (waveOutGetNumDevs())
	{
		printf("!!!ERROR!!! : FAILED TO OPEN 'MMSYSTEM' DEVICE\n");
		lErrorCount++;
	}
	else
	{
		printf("FAILED TO OPEN 'MMSYSTEM' DEVICE\n");
	}
}
printf("-------------------------------------\n\n");

////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
// TEST : Open NULL device
//

printf("------------------------------------\n");
printf("TESTING NULL DEVICE\n\n");
pDevice = alcOpenDevice(NULL);
if (pDevice)
{
	printf("OPENED NULL DEVICE ... GOT %s\n", alcGetString(pDevice,
ALC_DEVICE_SPECIFIER));
	alcCloseDevice(pDevice);
}
else
{
	if (waveOutGetNumDevs())
	{
		printf("!!!ERROR!!! : FAILED TO OPEN NULL DEVICE\n");
		lErrorCount++;
	}
	else
	{
		printf("FAILED TO OPEN NULL DEVICE\n");
	}
}
printf("------------------------------------\n\n");

////////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
// TEST : Open "" device
//

printf("------------------------------------\n");
printf("TESTING EMPTY DEVICE\n\n");
pDevice = alcOpenDevice("");
if (pDevice)
{
	printf("OPENED \"\" DEVICE ... GOT %s\n", alcGetString(pDevice,
ALC_DEVICE_SPECIFIER));
	alcCloseDevice(pDevice);
}
else
{
	if (waveOutGetNumDevs())
	{
		printf("!!!ERROR!!! : FAILED TO OPEN EMPTY DEVICE\n");
		lErrorCount++;
	}
	else
	{
		printf("FAILED TO OPEN EMPTY DEVICE\n");
	}
}
printf("------------------------------------\n\n");

////////////////////////////////////////////////////////////////////
```

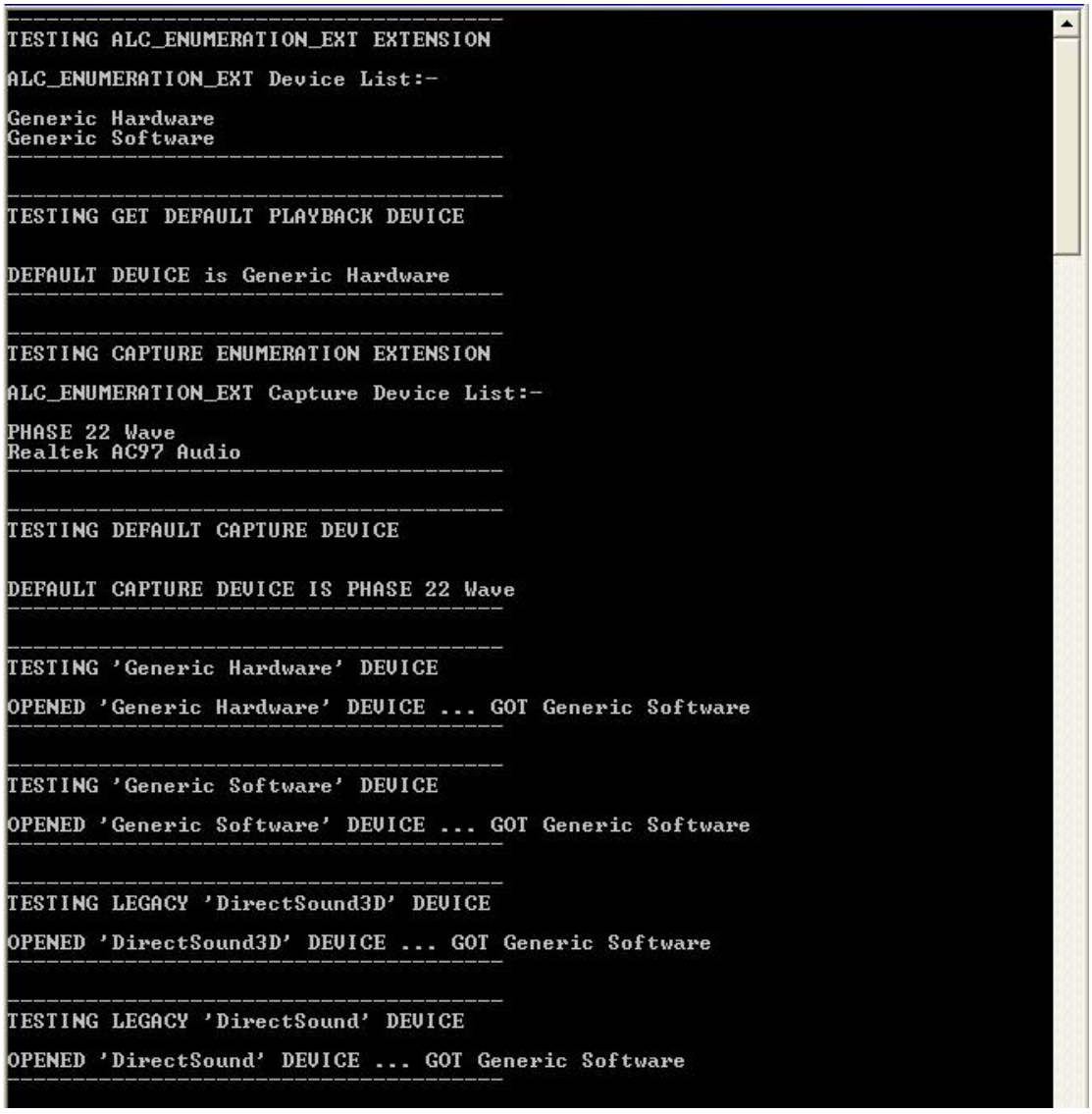
```
////////////////////////////////////////////////////////////////////
// TEST : Open "A Random Name" device
//
printf("-------------------------------------\n");
printf("TESTING 'A Random Name' DEVICE\n\n");
pDevice = alcOpenDevice("A Random Name");
if (pDevice)
{
	printf("!!!ERROR!!! : OPENED 'A Random Name' DEVICE ... GOT %s\n",
alcGetString(pDevice, ALC_DEVICE_SPECIFIER));
	lErrorCount++;
	alcCloseDevice(pDevice);
}
else
{
	printf("FAILED TO OPEN 'A Random Name' DEVICE\n");
}
printf("-------------------------------------\n\n");
////////////////////////////////////////////////////////////////////
```

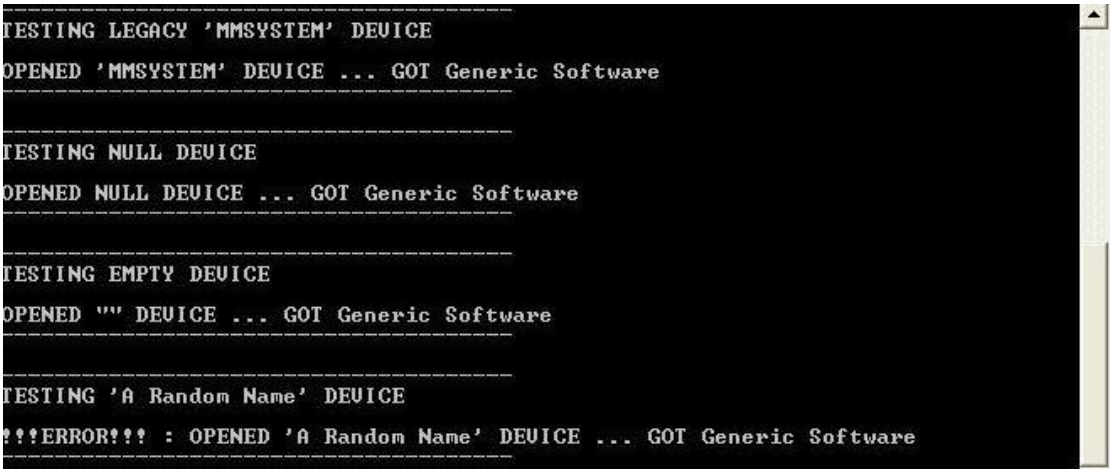
```
////////////////////////////////////////////////////////////////////
// TEST : Open NULL Capture device
//
printf("-------------------------------------\n");
printf("TESTING NULL CAPTURE DEVICE\n\n");
pDevice = alcCaptureOpenDevice(NULL, 22500, AL_FORMAT_MONO16, 4096);
if (pDevice)
{
	printf("OPENED NULL CAPTURE DEVICE ... GOT %s\n",
alcGetString(pDevice, ALC_CAPTURE_DEVICE_SPECIFIER));
	alcCaptureCloseDevice(pDevice);
}
else
{
	if (waveInGetNumDevs())
	{
		printf("!!!ERROR!!! : FAILED TO OPEN NULL CAPTURE DEVICE\n");
		lErrorCount++;
	}
	else
	{
		printf("FAILED TO OPEN NULL CAPTURE DEVICE\n");
	}
}
printf("-------------------------------------\n\n");
////////////////////////////////////////////////////////////////////
```

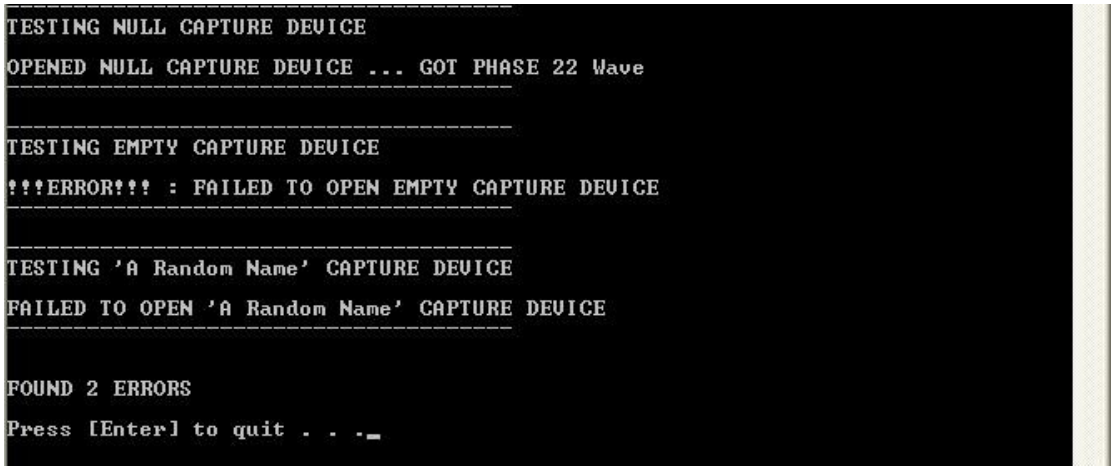
```
//////////////////////////////////////////////////////////////////////
// TEST : Open "" capture device
//
printf("----------------------------------------\n");
printf("TESTING EMPTY CAPTURE DEVICE\n\n");
pDevice = alcCaptureOpenDevice("", 22500, AL_FORMAT_MONO16, 4096);
if (pDevice)
{
	printf("OPENED \"\" CAPTURE DEVICE ... GOT %s\n",
alcGetString(pDevice, ALC_CAPTURE_DEVICE_SPECIFIER));
	alcCaptureCloseDevice(pDevice);
}
else
{
	if (waveInGetNumDevs())
	{
		printf("!!!ERROR!!! : FAILED TO OPEN EMPTY CAPTURE DEVICE\n");
		lErrorCount++;
	}
	else
	{
		printf("FAILED TO OPEN EMPTY CAPTURE DEVICE\n");
	}
}
printf("----------------------------------------\n\n");
//////////////////////////////////////////////////////////////////////
```

```
//////////////////////////////////////////////////////////////////////
// TEST : Open "A Random Name" capture device
//
printf("----------------------------------------\n");
printf("TESTING 'A Random Name' CAPTURE DEVICE\n\n");
pDevice = alcCaptureOpenDevice("A Random Name", 22500, AL_FORMAT_MONO16,
4096);
if (pDevice)
{
	printf("!!!ERROR!!! : OPENED 'A Random Name' CAPTURE DEVICE ... GOT
%s\n", alcGetString(pDevice, ALC_CAPTURE_DEVICE_SPECIFIER));
	lErrorCount++;
	alcCaptureCloseDevice(pDevice);
}
else
{
	printf("FAILED TO OPEN 'A Random Name' CAPTURE DEVICE\n");
}
printf("----------------------------------------\n\n");
//////////////////////////////////////////////////////////////////////

printf("\n FOUND %d ERRORS\n", lErrorCount);
printf ( "\n Press [Enter] to quit . . ." );
fflush ( stdout );
getchar();
return 0;
}
```

Τα αποτελέσματα του συγκεκριμένου παραδείγματος για ένα σύστημα με default κάρτα ήχου Phase 22 wave και έχοντας ενεργοποιημένη την onboard κάρτα ήχου της μητρικής (Realtek AC97 Audio) είναι :







Τα δύο Error που δημιουργήθηκαν είναι επιθυμητά, μιας και στις δύο περίπτωσεις το πρόγραμμα ζητάει λανθασμένη λειτουργία. Στην μια περίπτωση ζητάει το άνοιγμα ενός random name device, ενώ στην δεύτερη το άνοιγμα ενός empty capture device.

2. Example: Initialize / Exit

Στο συγκεκριμένο παράδειγμα γίνεται Initialize και στην συνεχεία Exit από την OpenAL.

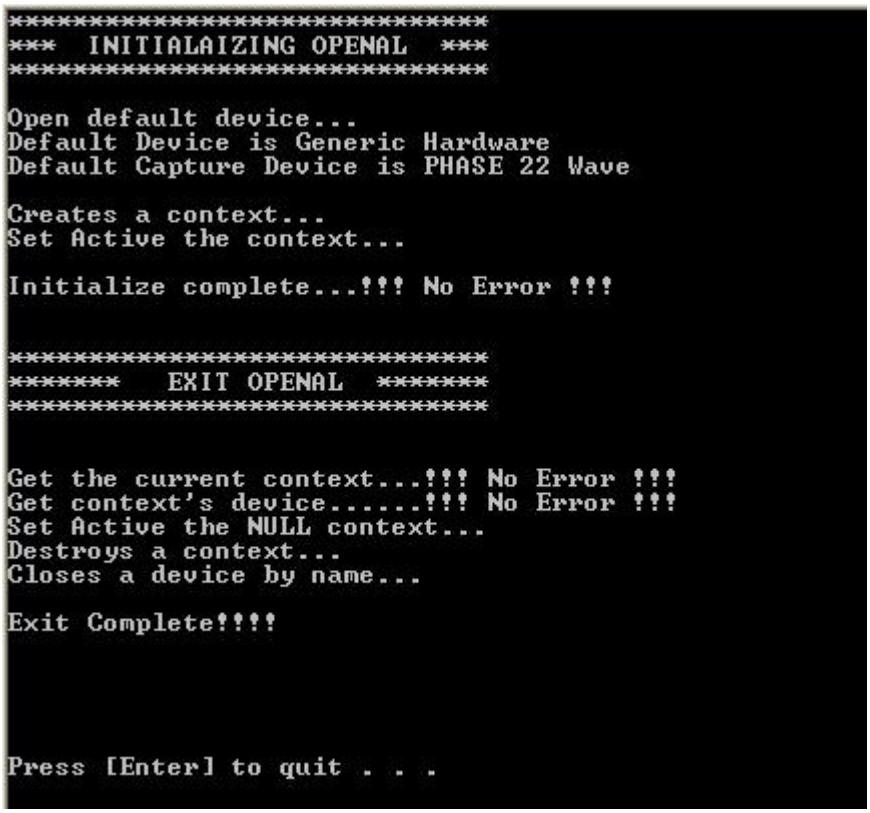
```
////////////////////////////////////////////////////////////
/*
 *                Initialization OpenAL!!!
 *
 *                1.Open default OpenAL device
 *                2.Creates a context
 *                3.Set active context
 *                4.Print ALC error code
 *
 *                Exit OpenAL!!!
 *
 *                1.Get the current context
 *                3.Get a context's device pointer
 *                4.Makes a NULL context the current context
 *                5.Destroys a context
 *                6.closes a device
 *
 */
////////////////////////////////////////////////////////////

/*Get some classic includes*/
#include <Windows.h>
#include <mmsystem.h>
#include <string.h>
#include <stdio.h>

/*includes FULL OpenAL*/
#include <al.h>
#include <alc.h>
#include <efx.h>
#include <efx-creative.h>
#include <xram.h>

int main()
{

/*Definition Variable*/
ALCdevice *pDevice = NULL;
ALCcontext *pContext = NULL;
ALCenum Error; /* save error status*/

printf ( "*****************************\n" );
printf ( "***  INITIALAIZING OPENAL  ***\n" );
printf ( "*****************************\n" );

/*Open default device*/
pDevice = alcOpenDevice(NULL);
if (pDevice != NULL) {
      printf ( "\nOpen default device...\n" );
      printf ( "Default Device is %s\n", alcGetString(pDevice,
                ALC_DEFAULT_DEVICE_SPECIFIER) );
      printf ( "Default Capture Device is %s\n",
```

```
alcGetString(pDevice, ALC_CAPTURE_DEFAULT_DEVICE_SPECIFIER) );

      alcGetError(pDevice);/*Clear error code*/

      /*creates a context using a default device*/
      pContext =alcCreateContext(pDevice,NULL);

      if (pContext != NULL) {
            printf ( "\nCreates a context..." );

      /*set active context-Makes a pContext the current context*/
            alcMakeContextCurrent(pContext);
            if (alcMakeContextCurrent(pContext)== ALC_TRUE){
                  printf ( "\nSet Active the context..." );
                  Error = alcGetError(pDevice);
                  if (Error != ALC_NO_ERROR){

               printf ( "\nERROR!!! %s\n", alGetString( Error ) );
                  }
                  else{
                        printf ( "\n\nInitialize complete...!!! %s
                                    !!! \n", alGetString( Error));
                  }
            }
            else {/*alcMakeContextCurrent(pContext)!= ALC_TRUE*/

                  printf ( "\nFAILED SET ACTIVE CONTEXT!\n" );
                  Error = alcGetError(pDevice);
                  if (Error != ALC_NO_ERROR){
                  printf ( "Unable Active OpenAL context %s\n",
                              alGetString( Error ) );
                  }
            }
      }
      else {/*if pContext = NULL*/
            printf ( "\nFAILED TO CREATES A CONTEXT!\n" );
            Error = alcGetError(pDevice);
            if (Error != ALC_NO_ERROR){
                  printf ( "Unable to create OpenAL context %s\n",
                              alGetString( Error ) );
            }
      }
}
else {/*pDevice = NULL*/
      printf ( "\nFAILED TO OPEN DEFAULT DEVICE!\n" );
      printf ( "No sound driver/device has been found!\n" );
      }


/**********************Initilize Code End ************************/
```

```
printf ( "\n\n" );
printf ( "*****************************\n" );
printf ( "*******   EXIT OPENAL  *******\n" );
printf ( "*****************************\n" );


alcGetError(pDevice);/*Clear resets the error state.*/

pContext = alcGetCurrentContext();/*Get the current context.*/
Error = alcGetError(pDevice);

if (pContext != NULL)/* If there is no current context, NULL is
returned*/
{
      printf ( "\n\nGet the current context..." );
      printf ( "!!! %s !!!", alGetString( Error ));
}
else{/*pContext == NULL*/
      printf ( "\n\nThere is no current context, NULL is
                  returned..." );
      printf ( "ERROR:!!! %s !!!", alGetString(Error ));
}

pDevice = alcGetContextsDevice(pContext);/*Get context's device
pointer*/
Error = alcGetError(pDevice);
if (Error != ALC_INVALID_CONTEXT)/*If the specified context is
invalid,ALC_INVALID_CONTEXT is returned*/
{
      printf ( "\nGet context's device......");
      printf ( "!!! %s !!!", alGetString( Error ));
}
else
{
      printf ( "\nThe specified context is invalid...");
      printf ( "ERROR:!!! %s !!!", alGetString( Error ));

}
alcMakeContextCurrent(NULL); /*Makes a specified context the current
context.*/
if (alcMakeContextCurrent(NULL)== ALC_TRUE)
{
      printf ( "\nSet Active the NULL context..." );
}
else
{
      printf ( "failure to Set Active the NULL context...\n" );
      printf ( "ERROR!!! ");
}
alcDestroyContext(pContext);/*destroys a context*/
printf ( "\nDestroys a context..." );
alcCloseDevice(pDevice);/*closes a device by name*/
printf ( "\nCloses a device by name...");

printf ( "\n\nExit Complete!!!!\n");

/*********************Exit Code End ************************/
```

```
printf ( "\n\n\n\n\nPress [Enter] to quit . . ." );
fflush ( stdout );
getchar();
return 0;

}
```

Τα αποτελέσματα του συγκεκριμένου παραδείγματος, σε ένα σύστημα που δεν θα δημιουργηθούν σφάλματα, είναι:



3. Example: Simple Static Sound

Στο παράδειγμα αυτό φαίνεται η πιο απλή διαδικασία εκτέλεσης ενός ήχου wav. Επίσης γίνεται Initialize και Exit από το API ALUT, συμπληρώνοντας έτσι το προηγούμενο παράδειγμα, που παρείχε το Initialize/Exit μόνο για την χρήση του με το AL API.

```
////////////////////////////////////////////////////////////////
/*
 *                      OpenAL SDK 1.1
 *                Example Code: 3.Simple Static Sound
 *
 *    1.Initializing/Exiting OpenAL and Alut
 *    2.Path sound file (....\\Code\\3. Example Simple Static
 *                      Sound\\Debug\\sounds)
 *    3.Load a sound file
 *    4.Generate a Source, Attach Source to Buffer, Play Source,
 *    5.Clean up by deleting Source and Buffer.
 *
 *
 */
////////////////////////////////////////////////////////////////
```

/*Include.h*/

```
#ifndef INCLUDE_H_
#define INCLUDE_H_
/*Get some classic includes*/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <Windows.h>
/*includes FULL OpenAL*/
#include <al.h>
#include <alc.h>
#include <alut.h>
#include <efx.h>
#include <efx-creative.h>
#include <xram.h>

/*Start program print*/
void StartPrint();
/*Quit program*/
void ExitEnterPress(void);

/*Initialization OpenAL*/
ALboolean InitOpenAL();
/*Exit-Close OpenAL*/
ALboolean CloseOpenAL();

/*Initialization ALUT*/
ALboolean InitALUT();
/*Exit-Close ALUT*/
ALboolean CloseALUT ();
```

```
/* Max. length of full pathname *stdlib.h VC++ LIB*/
#define MAXPATH   260
/* Path finder  for Sound */
ALchar *SoundPath(const ALchar *filename);
/* Load an AL buffer from the given sound file. */
ALuint  loadSound (const ALchar *testfile);
/*Generate a single source and sound playing*/
ALboolean SoundPlay(ALuint Buffer);

#endif /*INCLUDE_H_*/
```

/*Include.c*/

```
/*Include Function File*/
#include "include.h"
```

```
/*Start program print*/

void StartPrint()
{
printf ( "*****************************************************\n" );
printf ( "****************** OpenAL SDK 1.1 ******************\n" );
printf ( "******** Example Code:3.Simple Static Sound ********\n" );
printf ( "*****************************************************\n" );
return ;
}
```

```
/*quit program*/

void ExitEnterPress(void)
{
      printf ( "\n Press [Enter] to quit . . ." );
      fflush ( stdout );
      getchar();
      exit(0); /*Returns 0 to the operating system*/
}
```

```
////////////////////////////////////////////////////////////
/****************** Initialization OpenAL! *****************/

ALboolean InitOpenAL()
{

ALCcontext *pContext = NULL;
ALCdevice *pDevice = NULL;
ALboolean bReturn = AL_FALSE;
ALCenum Error; /* save error status*/
ALCboolean currentcon;/*test for alcMakeContextCurrent*/

printf ( "\n Initialization OpenAL............................" );

      pDevice = alcOpenDevice(NULL); /*Open default device*/
```

```
	if (pDevice){ /*pDevice != NULL*/
		alcGetError(pDevice);/*Clear error code*/
		/*creates a context using a default device*/
		pContext = alcCreateContext(pDevice, NULL);


		if (pContext){/*pContext != NULL*/
		/*set active - Makes a pContext the current context*/
			currentcon = alcMakeContextCurrent(pContext);
			if (currentcon == ALC_TRUE){
				printf ( "Done\n" );
				printf ( "  Default Device is %s\n",
					alcGetString(pDevice,
					ALC_DEFAULT_DEVICE_SPECIFIER));

				printf ( "  Default Capture Device is %s\n",
					 alcGetString(pDevice,
					ALC_CAPTURE_DEFAULT_DEVICE_SPECIFIER));
				bReturn = AL_TRUE;
			}
			else{
			 /*alcMakeContextCurrent(pContext)!= ALC_TRUE */
				Error = alcGetError(pDevice); /*error state*/
				printf ( "Error!!!:%s\n",
					alGetString( Error ));
				printf ( "Unable to make OpenAL context
					current.\n" );
				CloseOpenAL();
				ExitEnterPress();
			}
		}
		else {/*pContext = NULL*/
			Error = alcGetError(pDevice);/*error state*/
			printf ( "Error!!!:%s\n", alGetString( Error ));
			printf ( "Unable to create an OpenAL context.\n" );
			CloseOpenAL();
			ExitEnterPress();
		}
	}
	else{/*pDevice = NULL*/
		printf ( "Unable to Open default device.\n" );
		CloseOpenAL();
		ExitEnterPress();
	}
	return bReturn;
}
```

```
////////////////////////////////////////////////////////////
/******************** Exit-Close OpenAL! ********************/

ALboolean CloseOpenAL()
{

ALCcontext *pContext;
ALCdevice *pDevice;
ALCboolean currentcon ;/*test for alcMakeContextCurrent*/
```

```
printf ( "\n Exit OpenAL................................." );

pContext = alcGetCurrentContext(); /*open the current context */

if (alcGetCurrentContext() == NULL){
      printf ( "\n There is no current context, NULL is
                returned ........" );
      printf("\n Exit OpenAL....................................." );
      }

      /*open a context's device pointer*/
      pDevice = alcGetContextsDevice(pContext);

      /*makes NULL context the current context*/
      currentcon = alcMakeContextCurrent(NULL);

      if (currentcon != ALC_TRUE){
            printf ( "failure to makes NULL context the current
                      context......" );
            printf ( "\n Exit OpenAL.............................");
      }

      alcDestroyContext(pContext);/*destroys a context*/

      alcCloseDevice(pDevice);/*closes a device*/

      printf ( "Done\n" );

      return AL_TRUE;
}
```

```
//////////////////////////////////////////////////////////////
/****************** Initialization ALUT!! *****************/
/*** Info: Openal -svn\trunk\OpenAL-Sample\test\testlib.c ***/

ALboolean  InitALUT()
{
      ALboolean   bReturn = AL_FALSE;
      ALenum            Error; /*Save ALUT error status*/
      printf ( "Initialization Alut..........................." );

      alutGetError();/*Clear error code*/

      if (!alutInitWithoutContext( NULL , NULL )) {
            Error = alutGetError();

            printf("Error: %s\n", alutGetErrorString(Error));
            CloseALUT();
            CloseOpenAL();
            ExitEnterPress();

      }
      printf ( "Done" );
      bReturn = AL_TRUE;
      return bReturn;
}
```

```
////////////////////////////////////////////////////////////
/******************** Exit-Close ALUT!! ******************/
/*** Info: Openal -svn\trunk\OpenAL-Sample\test\testlib.c ***/

ALboolean CloseALUT()
{
      ALboolean   bReturn = AL_FALSE;
      ALenum            Error;      /*Save ALUT error status*/
      printf ( "\n Exit Alut....................................." );

      alutGetError();/*Clear error code*/

      if (!alutExit()) {
            Error = alutGetError();
            printf("Error: %s\n", alutGetErrorString(Error));
            CloseALUT();
            CloseOpenAL();
            ExitEnterPress();
      }
      printf ( "Done" );
      bReturn = AL_TRUE;
      return bReturn;
}
```

```
////////////////////////////////////////////////////////////
/********************** Path finder ************************/

ALchar fullPath[MAXPATH];
ALchar *SoundPath(const ALchar *filename)// Path Finder For Sound
{
      sprintf(fullPath,"sounds\\%s",filename);
      return fullPath;
}
```

```
////////////////////////////////////////////////////////////
/********************* Load a sound ************************/
/***** Create an AL buffer from the given sound file. ******/

ALuint  loadSound ( const ALchar *testfile )
{
      ALenum            Error; /*Save AL and ALUT error status*/
      ALchar         *FileName;/*Full Path sound file */
      ALuint            Buffer;/*Buffer Name */

alGetError();     /*Clear AL error code*/

alutGetError();   /*Clear ALUT error code*/

printf ( "\n Load Sound File....................................." );

      /*Full Path sound file*/
      FileName = SoundPath(testfile);

      if (!FileName){
            printf("Error!!! Unable to Open File: %s\n",
                  SoundPath(testfile));
```

```
            CloseALUT();
            CloseOpenAL();
            ExitEnterPress();
      }
      else {


            /*load a sound file into an OpenAL buffer*/
            Buffer = alutCreateBufferFromFile ( FileName );
            if (Buffer == AL_NONE )
            {
                  Error = alutGetError();
                  printf ( "\n Error loading file: %s\n",
                           alutGetErrorString (Error));

                  printf("Failed to load %s\n", SoundPath(testfile));
                  alDeleteBuffers(1,&Buffer);/*Delete buffer*/
                  CloseALUT();
                  CloseOpenAL();
                  ExitEnterPress();
            }
      }
      printf ( "Done\n\n" );
      return Buffer;
}
```

```
//////////////////////////////////////////////////////////////
/******************** Sound play ************************/
/****** Generate a single source and sound playing ********/

ALboolean SoundPlay(ALuint Buffer)
{

      ALenum            Error; /*Save AL and ALUT error status*/
      ALuint            Sources;/*sources Name */
      ALint             status;/*Source State*/
      ALboolean   bReturn = AL_FALSE;

      alGetError();     /*Clear AL error code*/

      alGenSources (1, &Sources);/*generates one source*/

      Error = alGetError ();
      if ( Error != AL_NO_ERROR )
       {
         printf ( "!!Error!! Failure to Generate a Source.
                   !!Error!!: %s\n", alGetString (Error));
          alDeleteSources(1, &Sources);
          alDeleteBuffers(1,&Buffer);
          CloseALUT();/*Exit-Close ALUT*/
          CloseOpenAL();/*Exit-Close OpenAL*/
          ExitEnterPress();/*quit program*/
          }

      /*Attach Source to Buffer*/
      /*sets an integer property of a source*/
      alSourcei (Sources, AL_BUFFER, Buffer);
```

```
      Error = alGetError ();
      if ( Error != AL_NO_ERROR )
       {
          printf ( "!!Error!! Failure to Attach Source to Buffer.
                    !!Error!!: %s\n", alGetString (Error));
          alDeleteSources(1, &Sources);
          alDeleteBuffers(1,&Buffer);
          CloseALUT();/*Exit-Close ALUT*/
          CloseOpenAL();/*Exit-Close OpenAL*/
          ExitEnterPress();/*quit program*/
          }

      alSourcePlay (Sources);/*plays a source*/
      Error = alGetError ();
      if ( Error != AL_NO_ERROR )
       {
         printf ( "!!Error!! Failure to Play Source.
                   !!Error!!: %s\n", alGetString (Error));
         alSourceStop(Sources);/*stops a source*/
         alDeleteSources(1, &Sources);
         alDeleteBuffers(1,&Buffer);
          CloseALUT();/*Exit-Close ALUT*/
          CloseOpenAL();/*Exit-Close OpenAL*/
          ExitEnterPress();/*quit programme*/
          }

      printf("Playing Source ");

        do
          {
            Sleep(100);
            printf (".");
            /* Get Source State*/
            alGetSourcei (Sources, AL_SOURCE_STATE, &status);
          }
        while (status == AL_PLAYING);
        printf ( "...Done\n\n\n" );

/*Clean up by deleting Source and Buffer*/
        alSourceStop(Sources);
        alDeleteSources(1, &Sources);
        alDeleteBuffers(1,&Buffer);
        bReturn = AL_TRUE;
        return bReturn;
}
```

/* Main.c*/

```
#include "include.h"

int main()
{
ALuint      Buffer;                          /*Buffer Name */
ALchar      TestFileName [FILENAME_MAX]; /*260 stdio.h ONLY MinGW*/

//////////////////////////////////////////////////////////////
/********** Start program and Initialization code **********/

      StartPrint(); /*Start program print*/
      InitOpenAL(); /*Initialization OpenAL*/
      InitALUT(); /*Initialization OpenAL ALUT*/

//////////////////////////////////////////////////////////////
/********************** Main program **********************/
printf ("\n\n\n Default Sound Path is Code\\3. Example Simple Static
Sound\\Debug\\sounds");
printf ("\n Enter your Sound File Name: ");
fgets(TestFileName, sizeof(TestFileName), stdin);
/*Remove the new line character (remove the Enter)*/
strtok(TestFileName, "\n");
/* Load an AL buffer from the given sound file. */
Buffer = loadSound (TestFileName);
/* Generate a single source and start playing. */
         SoundPlay (Buffer);
//////////////////////////////////////////////////////////////
/***************** Exit Code - Exit Program *****************/
      alDeleteBuffers(1,&Buffer);
      CloseALUT();/*Exit-Close ALUT*/
      CloseOpenAL();/*Exit-Close OpenAL*/
      ExitEnterPress();/*quit program*/
      return 0;

}
```

Ο χρήστης μπορεί να εκτελέσει έναν ήχο wav που θα επιλέξει:

4. Example: Looping

Στο παράδειγμα ο χρήστης έχει την δυνατότητα να δει τα βασικά στοιχεία του ήχου που μόλις φόρτωσε (frequency, Channel, Bits, Size), ενώ στη συνέχεια μπορεί να εκτελέσει τον ήχο, να τον θέσει σε κατάσταση Loop και εντέλει, όταν επιθυμεί, να τον σταματήσει. Κάθε φορά που εκτελείται το play, θέτει το Loop AL_FALSE, ώστε να μπορεί να σταματήσει ο ήχος όταν ολοκληρωθεί η ανάγνωση από τον buffer. Όλο το πρόγραμμα βασίζεται σε ένα πολύ απλό interface, που χρησιμοποιεί τα πλήκτρα του πληκτρολόγιου (play είναι το 1, Loop on το 2, stop είναι το 3 και q είναι το quit.). Επιπρόσθετα τα errors, αν και εφόσον υπάρξουν, εκτυπώνονται σε κόκκινη μορφή.

```
/*
 *                OpenAL SDK 1.1
 *           Example Code: Looping
 *     1.Initializing & exiting OpenAL and Alut
 *     2.User set the Path sound file
 *     3.Print red colour error code
 *     4.Load a .wav sound
 *     5.Print Sound info(Frequency, Channel, Bits and Size)
 *     6.Create an AL buffer from the given sound file
 *     7.Generate a Source, Attach Source to Buffer, Play Source,
 *       Loop Source ,Stop Source.
 *     8.Clean up by deleting Source and Buffer
 *
 *
 */
```

/*Include.h*/

```
#ifndef INCLUDE_H_
#define INCLUDE_H_

/*Get some classic includes*/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <Windows.h>
/*This include work ONLY MinGW. NOT compiler work Cygwin...*/
#include <conio.h>

/*includes FULL OpenAL*/
#include <al.h>
#include <alc.h>
#include <alut.h>
#include <efx.h>
#include <efx-creative.h>
#include <xram.h>

/************** GENERAL FUNCTION **************/
void whitePrint(const char *Text);/*Print White Text*/
void RedPrint(const char *Text);/*Print Red  Text */

/*Print Red AL-ALC Error code & Text */
void RedALErrorPrint(const char *Text, ALenum Error);
```

```
/*Print Red ALUT Error code & Text */
void RedALUTErrorPrint(const char *Text, ALenum Error);

/* Path finder  for Sound */
ALchar *SoundPath(const ALchar *filename, const ALchar *Path);

/*Use sound Path Function and add the user sound Path*/
ALchar *UserSoundPath();

void StartPrint();/*Start program print*/
void ExitEnterPress(void); /*quit programme*/

/****** INITIALIZATION & EXIT FUNCTION ******/
ALboolean InitOpenAL();/*Initialization OpenAL*/
ALboolean CloseOpenAL();/*Exit-Close OpenAL*/
ALboolean InitALUT();/*Initialization ALUT*/
ALboolean CloseALUT ();/*Exit-Close ALUT*/


/************* SOUND FUNCTION *************/
ALuint  LoadWavSound ();/*Load a .WAV sound -Create an AL buffer
from the given sound file.*/

#endif /*INCLUDE_H_*/
```

/*Include.c*/

```
/*Include Function File*/
#include "include.h"

/* Colour info:
 * Win32 Console Applications Tutorials- Part 4 colour.
 * http://www.adrianxw.dk/SoftwareSite/Consoles/Consoles4.html*/
```

```
/************** GENERAL FUNCTION **************/

void StartPrint(){/*Start program print*/

      HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
      SetConsoleTextAttribute(h, FOREGROUND_RED | FOREGROUND_GREEN |
                              FOREGROUND_BLUE |FOREGROUND_INTENSITY);

printf ( "*************************************************\n" );
printf ( "****************** OpenAL SDK 1.1 ******************\n" );
printf ( "************** Example Code:4.Looping **************\n" );
printf ( "*************************************************\n" );
      return ;
}
```

```
void whitePrint(const char *Text) {/*Print White Text*/

      HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
      SetConsoleTextAttribute(h, FOREGROUND_RED | FOREGROUND_GREEN |
                              FOREGROUND_BLUE |FOREGROUND_INTENSITY);
      printf("%s", Text);
}
```

```
void RedPrint(const char *Text) {/*Print Red  Text */

	HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
	SetConsoleTextAttribute(h, FOREGROUND_RED
				|FOREGROUND_INTENSITY);
	printf("%s", Text);
}
```

```
/*Print Red AL-ALC Error code & Text */
void RedALErrorPrint(const char *Text, ALenum Error) {

	HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
	SetConsoleTextAttribute(h, FOREGROUND_RED
				 |FOREGROUND_INTENSITY);
	printf("%s%s\n", Text, alGetString(Error));
}
/*Print Red ALUT Error code & Text */
void RedALUTErrorPrint(const char *Text, ALenum Error) {

	HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
	SetConsoleTextAttribute(h, FOREGROUND_RED
				|FOREGROUND_INTENSITY);
	printf("%s%s\n", Text, alutGetErrorString(Error));
}
```

```
void ExitEnterPress(void){/*quit programme*/
	whitePrint ( "\n Press [Enter] to quit . . ." );
	fflush ( stdout );
	getchar();
	exit(0); /*Returns 0 to the operating system*/
}
```

```
/*  Define MAX_PATH  = 260 stdlib.h This work ONLY MinGW.
 *  NOT define MAX_PATH  = 260 stdlib.h Cygwin */
ALchar fullPath[MAX_PATH];
/* Path finder  for Sound */
ALchar *SoundPath(const ALchar *filename, const ALchar *Path) {

	sprintf(fullPath, "%s%s", Path ,filename);
	return fullPath;
}
```

```
/*Use sound Path Function and add the user sound Path*/
ALchar *UserSoundPath(){

ALboolean AReturn = AL_FALSE;/* TRUE OR FALSE Function Return*/
ALchar    Path [MAX_PATH];   /*260 stdio.h ONLY MinGW*/
ALchar    *FileName;         /*Full Path sound file */
ALchar    TestFileName [ FILENAME_MAX ];  /*260 stdio.h ONLY MinGW*/
ALchar    ch;

	sprintf(Path, "sounds\\");/*Default Sound Path*/
	whitePrint("\n\n Default Sound File is ...");
	printf("%s", Path);
	do
	{
	printf("\n You Would Want To fix New Default Sound Path
		?(y/n)");
	ch = _getch();
```

```
            if(ch == 'y')
            {
                  printf ("\n Enter Your New Default Sound Path: ");
                  fgets(Path, sizeof(Path), stdin);
                /* Remove the new line character(remove the Enter)*/
                  strtok(Path, "\n");
                  printf ("Enter your Sound File Name: ");
                  fgets(TestFileName, sizeof(TestFileName), stdin);
               /* Remove the new line character(remove the Enter)*/
                  strtok(TestFileName, "\n");
            }
            else if (ch == 'n')
            {
                  printf ("\n Enter your Sound File Name: ");
                  fgets(TestFileName, sizeof(TestFileName), stdin);
               /* Remove the new line character(remove the Enter)*/
                  strtok(TestFileName, "\n");                }
            else
            {
            printf ("\n Error Character! Press y or n\n");
            }

            FileName = SoundPath(TestFileName, Path);
            printf ("Full Sound Path Name is :... %s", FileName);
            printf (" Correct?(y/n) ");
            ch = _getch();
            if(ch == 'y')
            {
                  AReturn = AL_TRUE;
            }

      } while (AReturn != AL_TRUE);

      return FileName;
}
```

```
/****** INITIALIZATION & EXIT FUNCTION ******/

/*Initialization OpenAL manually*/
ALboolean InitOpenAL(){

ALCcontext        *pContext = NULL;
ALCdevice         *pDevice = NULL;
ALboolean          bReturn = AL_FALSE;
ALenum             Error; /* save error status*/
ALCboolean         currentcon ;/*test for alcMakeContextCurrent*/


pDevice = alcOpenDevice(NULL); /*Open default device*/
if (pDevice){ /*pDevice != NULL*/
      alcGetError(pDevice);/*Clear error code*/
      /*creates a context using a default device*/
      pContext = alcCreateContext(pDevice, NULL);
      if (pContext){/*pContext != NULL*/
      /*set active context-Makes a pContext the current context*/
            currentcon = alcMakeContextCurrent(pContext);
            if (currentcon == ALC_TRUE){
                  bReturn = AL_TRUE;
            }
```

```
            else{/*alcMakeContextCurrent(pContext)!= ALC_TRUE */
                 /*context error state*/
                  Error = alcGetError(pDevice);
                  RedALErrorPrint( "Unable to make OpenAL context
                                    current: !!!Error!!!", Error );
            }
      }
      else {/*pContext = NULL*/
            Error = alcGetError(pDevice);/*context error state*/
            RedALErrorPrint ( "Unable to create an OpenAL context:
                              !!!Error!!!", Error);
      }
}
else{/*pDevice = NULL*/

HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
            SetConsoleTextAttribute(h, FOREGROUND_RED
                                    |FOREGROUND_INTENSITY);
            printf("Unable to Open default device :!!!Error!!!");

}
return bReturn;
}
```

```
ALboolean CloseOpenAL(){/*Exit-Close OpenAL*/
ALCcontext *pContext;
ALCdevice *pDevice;
ALCboolean currentcon ;/*test for alcMakeContextCurrent*/

pContext = alcGetCurrentContext(); /*open the current context */
if (alcGetCurrentContext() == NULL){
      whitePrint ( "\n WARNING: There is no current context, NULL is
                    returned........" );
}
/*open a context's device pointer*/
pDevice = alcGetContextsDevice(pContext);
/*makes NULL context the current context*/
currentcon = alcMakeContextCurrent(NULL);
if (currentcon != ALC_TRUE){
      whitePrint ( "WARNING: failure to makes NULL context the
                    current context......" );
}
alcDestroyContext(pContext);/*destroys a context*/
alcCloseDevice(pDevice);/*closes a device*/
return AL_TRUE;
}
```

```
ALboolean  InitALUT(){/*Initialization ALUT*/
ALboolean   bReturn = AL_FALSE;
ALenum      Error; /*Save ALUT error status*/

      alutGetError();/*Clear error code*/
      if (!alutInitWithoutContext( NULL , NULL )) {
            Error = alutGetError();
            RedALUTErrorPrint("!!!Error!!!", Error);
      }
      else {
           bReturn = AL_TRUE;
      }
      return bReturn;
}
```

```
ALboolean CloseALUT(){/*Exit-Close ALUT */

ALboolean   bReturn = AL_FALSE;
ALenum      Error;      /*Save ALUT error status*/
      alutGetError();/*Clear error code*/

      if (!alutExit()) {
            Error = alutGetError();
            RedALUTErrorPrint("Close ALUT : !!!Error!!!", Error);
      }
      bReturn = AL_TRUE;
      return bReturn;
}
```

```
/************* SOUND FUNCTION **************/

/*Load a .WAV sound -Create an AL buffer from the given sound file*/
ALuint  LoadWavSound (){

ALenum   Error;         /*Save AL and ALUT error status*/
ALchar  *FileName;      /*Full Path sound file */
ALuint   Buffer;        /*Buffer Name */

alGetError();     /*Clear AL error code*/
alutGetError();   /*Clear ALUT error code*/

whitePrint("\n\n\n Load Wav Sound File:");
FileName = UserSoundPath();/*Full Path sound file*/

if (!FileName){
      RedPrint("\n!!! Error !!! Unable to Open File:");
      printf("%s\n", FileName);
      CloseALUT();/*Exit-Close ALUT*/
      CloseOpenAL();/*Exit-Close OpenAL*/
      ExitEnterPress();/*quit programme*/
}
else {
      Buffer = alutCreateBufferFromFile ( FileName );

      if (Buffer == AL_NONE )
      {
            Error = alutGetError();
            RedALUTErrorPrint("\n!!! Error!!! loading file:",Error);
            printf("Failed to load %s\n", FileName);
            alDeleteBuffers(1,&Buffer);/*delete Buffer*/
            CloseALUT();/*Exit-Close ALUT*/
            CloseOpenAL();/*Exit-Close OpenAL*/
            ExitEnterPress();/*quit programme*/
      }
}

return Buffer;
}
```

/* Main.c*/

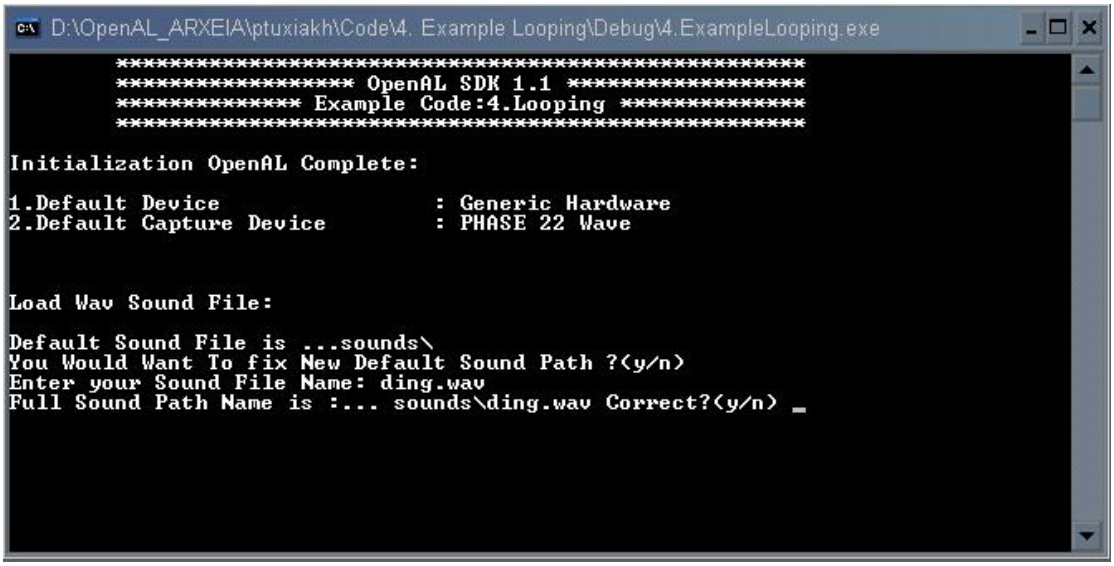
```
/*Include Function File*/
#include "include.h"


int main ( void )
{

/****************** Definition Variable *****************/
ALboolean   AReturn  = AL_FALSE;/* TRUE OR FALSE Function Return*/
ALboolean   BReturn  = AL_FALSE;/* TRUE OR FALSE Function Return*/
ALCdevice   *pDevice = NULL;
ALenum      Error;              /*Save AL and ALUT error status*/
ALuint      Buffer;             /*Buffer Name */
ALuint      Sources;            /*sources Name */
ALbyte      ch;                 /*character for while */
ALint       channels;           /*Sound Channel info for buffer*/
ALsizei     freq;               /*Sound Frequency info for buffer*/
ALint       bits;               /*Sound Bits info for buffer*/
ALsizei     size;               /*Sound size info for buffer*/

/************ INITIALIZATION OpenAL AND ALUT ************/

StartPrint();/*Start program print*/

AReturn   = InitOpenAL();
BReturn   = InitALUT();

if ( AReturn == AL_TRUE && BReturn == AL_TRUE ) {

       whitePrint( "\n Initialization OpenAL Complete:\n\n" );
        printf( "1.Default Device          : %s\n",
           alcGetString(pDevice,ALC_DEFAULT_DEVICE_SPECIFIER));

        printf( "2.Default Capture Device : %s\n",
       alcGetString(pDevice, ALC_CAPTURE_DEFAULT_DEVICE_SPECIFIER));
}
else {

      RedPrint("\n !ERROR! Initialization OpenAL NOT Complete\n\n");
      CloseALUT();
      CloseOpenAL();
      ExitEnterPress();
}

/************** MAIN PROGRAM-SOUND  PLAY **************/

/*Load a .WAV sound -Create an AL buffer from the given sound file*/
Buffer = LoadWavSound ( );

whitePrint( "\n\n\n Load Sound info:\n" );
/*retrieves Frequency info for buffer*/
alGetBufferi(Buffer, AL_FREQUENCY, &freq);
/*retrieves Bits info for buffer*/
alGetBufferi(Buffer, AL_BITS, &bits);
/*retrieves Channel info for buffer*/
alGetBufferi(Buffer, AL_CHANNELS, &channels);
/*retrieves size info for buffer*/
alGetBufferi(Buffer, AL_SIZE, &size);
```

```
printf("\n Frequency %iHz", freq);
printf("\n Channel   %i", channels);
printf("\n Bits      %i", bits);
printf("\n Size      %i", size);

alGetError();     /*Clear AL error code*/

/*Generate a single source*/
alGenSources (1, &Sources);
Error = alGetError ();
if ( Error != AL_NO_ERROR ){

     RedALErrorPrint("\n!!! Error !!!Failure to Generate a Source:",
                     Error);
      alDeleteSources(1, &Sources);/*delete source*/
      alDeleteBuffers(1,&Buffer); /*delete Buffer*/
      CloseALUT();/*Exit-Close ALUT*/
      CloseOpenAL();/*Exit-Close OpenAL*/
      ExitEnterPress();/*quit programme*/
}

/*Attach Source to Buffer*/
alSourcei (Sources, AL_BUFFER, Buffer);
Error = alGetError ();
if ( Error != AL_NO_ERROR ){

      RedALErrorPrint("\n!!! Error !!!Failure to Attach Source to
                      Buffer:", Error);
      alDeleteSources(1, &Sources);/*delete source*/
      alDeleteBuffers(1,&Buffer); /*delete Buffer*/
      CloseALUT();/*Exit-Close ALUT*/
      CloseOpenAL();/*Exit-Close OpenAL*/
      ExitEnterPress();/*quit programme*/
}

whitePrint("\n\n\n Select a test from the following options:\n\n");
whitePrint("Press '1' to play source\n");
whitePrint("Press '2' to toggle looping on source \n");
whitePrint("Press '3' to stop source \n");
whitePrint("Press 'q' to quit\n");
printf("\n\n\n\n\n\n\n\n\n");

do{
      ch = _getch();
      if (ch == '1'){

            alSourcei(Sources, AL_LOOPING, AL_FALSE );/*Loop OFF*/
            alSourcePlay (Sources);/*play a source*/
            if ( Error != AL_NO_ERROR ){

                  RedALErrorPrint("\n!!! Error !!!Failure to Play
                                  Source:", Error);
                  alSourceStop(Sources);/*stopped source*/
                  alDeleteSources(1, &Sources);/*delete source*/
                  alDeleteBuffers(1,&Buffer); /*delete Buffer*/
                  CloseALUT();/*Exit-Close ALUT*/
                  CloseOpenAL();/*Exit-Close OpenAL*/
                  ExitEnterPress();/*quit programme*/
            }
```

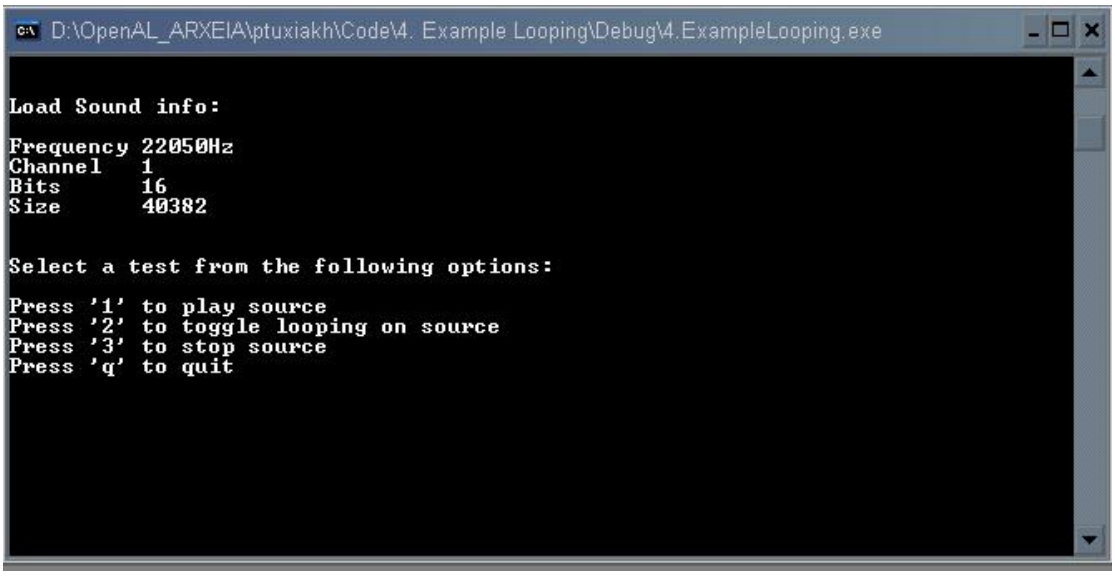
```
	}

	else if (ch == '2'){

		alSourcei(Sources, AL_LOOPING, AL_TRUE);/*Loop ON*/
		if ( Error != AL_NO_ERROR ){

			RedALErrorPrint("\n!!! Error !!!Failure to Loop
					Source:", Error);
			alSourceStop(Sources);/*stopped source*/
			alDeleteSources(1, &Sources);/*delete source*/
			alDeleteBuffers(1,&Buffer); /*delete Buffer*/
			CloseALUT();/*Exit-Close ALUT*/
			CloseOpenAL();/*Exit-Close OpenAL*/
			ExitEnterPress();/*quit programme*/
		}
	}
	else if (ch == '3')
	{
		alSourceStop(Sources);/*stopped source*/
		if ( Error != AL_NO_ERROR ){

			RedALErrorPrint("\n!!! Error !!!Failure to Stop
					Source: ",Error);
			alSourceStop(Sources);/*stopped source*/
			alDeleteSources(1, &Sources);/*delete source*/
			alDeleteBuffers(1,&Buffer); /*delete Buffer*/
			CloseALUT();/*Exit-Close ALUT*/
			CloseOpenAL();/*Exit-Close OpenAL*/
			ExitEnterPress();/*quit programme*/
		}
	}

}while (ch != 'q');


/*************** EXIT CODE - EXIT PROGRAM ***************/

/*Clean up by deleting Source and Buffer*/
alSourceStop(Sources);/*stopped source*/
alDeleteSources(1, &Sources);/*delete source*/
alDeleteBuffers(1,&Buffer); /*delete Buffer*/

CloseALUT();/*Exit-Close ALUT*/
CloseOpenAL();/*Exit-Close OpenAL*/
whitePrint( "\n Exit OpenAL Complete" );
ExitEnterPress();/*quit programme*/
return 0;

}
```

Επομένως, βάσει του παραπάνω κώδικα, ο χρήστης αρχικά θα επιλέξει τον ήχο που επιθυμεί να εκτελέσει και στην συνεχεία θα μπορεί να παίξει, σταματήσει ή να τον θέσει σε Loop. Ενδιαφέρον παρουσιάζει το πολλαπλό πάτημα του πλήκτρου 1 μιας και επανεκκινεί κάθε φορά τον ήχο, παρόλο που βρίσκεται ήδη σε διαδικασία AL_PLAYING.

Ο χρήστης μπορεί να επιλέξει τον ήχο που επιθυμεί να διαχειριστεί και



στην συνέχεια, μέσω ενός απλού interface, τον διαχειρίζεται.



5. Example: Multiple Sources

Το συγκεκριμένο παράδειγμα παρουσιάζει ορισμένες από τις βασικές δυνατότητες του πυρήνα της OpenAL. Αποτελεί ένα πλήρες ηχητικό περιβάλλον για τον χρήστη, ενώ ταυτόχρονα του δίνει την δυνατότητα να χειριστεί ορισμένες παραμέτρους.

Το πρόγραμμα φορτώνει ένα πλήθος από buffers. Οι πρώτοι οκτώ buffers χρησιμοποιούνται με την λειτουργία Queue Buffers, θέτοντας ένα background που αποτελείται από οκτώ μουσικά κομμάτια. Η λειτουργία Queue Buffers στο συγκεκριμένο παράδειγμα χρησιμοποιείται έτσι ώστε να θέσει σε ουρά αναμονής τα οκτώ κομμάτια σε ένα συγκεκριμένο Source. Το σύνολο των οκτώ buffers αντιστοιχεί σε 20:05 λεπτά (ή 210 MB, κάτι που αποτελεί έναν αρκετά μεγάλο όγκο πληροφορίας), έχοντας ως αποτέλεσμα, μέσω της λειτουργίας Queue Buffers, την ελαχιστοποίηση της χρήσης του CPU από το πρόγραμμα, αλλά ταυτόχρονα την αύξηση της χρησιμοποιούμενης μνήμης, μιας και τα οκτώ κομμάτια προετοιμάζονται και φορτώνονται στο Source (μνήμη), ώστε να είναι έτοιμα προς εκτέλεση όταν έρθει η σειρά τους.

Όπως έχει προαναφερθεί, η λειτουργία Queue Buffers μπορεί να χρησιμοποιηθεί με πολλαπλούς τρόπους. Για την ελαχιστοποίηση της χρησιμοποιούμενης μνήμης θα μπορούσαμε, μέσω των ιδιοτήτων που παρέχει η AL, να ελέγχουμε τον buffer που εκτελείται την συγκεκριμένη στιγμή και να προετοιμάζουμε τον επόμενο για εκτέλεση, θέτοντάς τον σε ουρά αναμονής, ενώ ταυτόχρονα θα μπορούσαμε να ελευθερώνουμε αυτόν που έχει ήδη εκτελεστεί, δημιουργώντας έναν βρόχο. Ο τρόπος που θα χειρισθεί ο προγραμματιστής την λειτουργία Queue Buffers, έχει άμεση σχέση με τους πόρους του συστήματος που χρησιμοποιεί το σύνολο της εφαρμογής που θέλει να δημιουργήσει. Μια εφαρμογή που χρειάζεται το σύνολο του CPU για την λειτουργία της, θα προτιμήσει οι buffers να είναι ήδη προετοιμασμένοι και φορτωμένοι στην μνήμη, ενώ μια εφαρμογή που χρειάζεται περισσότερη μνήμη θα προτιμήσει την λειτουργία που προαναφέραμε.

Στο συγκεκριμένο παράδειγμα οι οκτώ Queue Buffers τίθενται σε ουρά αναμονής και το Source τους τίθεται σε Loop ON. Ο χρήστης, μέσω του επάνω και κάτω βέλους του πληκτρολόγιου, μπορεί να αυξομειώσει το gain του Source, ενώ επειδή δεν υπάρχει κάποια «γραφική» ένδειξη για την ελάχιστη και την μέγιστη ένταση, χρησιμοποιείται μια ηχητική ένδειξη, που εκτελείται όταν το gain του Source είναι στο μέγιστο ή στο ελάχιστο. Η συγκεκριμένη ηχητική ένδειξη μπορεί να ακουστεί και όταν πατηθεί το πλήκτρο backspace.

Εκτός από την λειτουργία Queue Buffers, το συγκεκριμένο παράδειγμα παρουσιάζει και την λειτουργία της χωροτοποθέτησης του πυρήνα της AL.

Ο Listener τίθεται στις default τιμές AL_POSITION και AL_ORIENTATION (πρέπει να σημειωθεί ότι οι συγκεκριμένες γραμμές κώδικα θα μπορούσαν να παραληφθούν και παρουσιάζονται καθαρά για λόγους κατανόησης του κώδικα). Τέσσερα ακόμα Sources χρησιμοποιούνται για την δημιουργία του περιβάλλοντα χώρου.

Ο πρώτος Source είναι τοποθετημένος κοντά-μπροστά από τον χρήστη στο σημείο (0,0,0) και μπορεί να εκτελεστεί όταν ο χρήστης πατήσει το Spacebar.

Ο δεύτερος είναι τοποθετημένος μπροστά από τον χρήστη, αλλά σε κάποια απόσταση (0,0,1) και είναι σε Loop ON. Ο συγκεκριμένος ήχος διαθέτει ένα πλήρες περιβάλλον με ανακλάσεις, ώστε να δίνεται η αίσθηση του χώρου και παρόλο που δεν μπορεί να επηρεασθεί από τον χρήστη και έχει αρκετά χαμηλή ένταση, επηρεάζεται μέσω του gain των Queue Buffers, μιας και όταν μειώνεται αυτός γίνεται πλήρως εμφανής, ενώ όταν αυξάνεται λειτουργεί ως ηχητικό υπόβαθρο.

Ο τρίτος ήχος είναι και αυτός τοποθετημένος σε κάποια απόσταση, αλλά και κάτω από τον χρήστη. Εκτελεί την λειτουργία του Panning. Όπως έχει προαναφερθεί, το panning στην OpenAL λειτουργεί έμμεσα. Ο συγκεκριμένος ήχος τοποθετείται μια φορά δεξιά–μπροστά–κάτω (2,-4,-4) και μια φορά αριστερά–μπροστά–κάτω (-2,-4,-4). Κάθε φορά ελέγχεται το status AL_PLAYING και μία flag, που ονομάζεται Left. Αν ο ήχος δεν είναι σε κατάσταση AL_PLAYING και το flag Left είναι αληθές, τότε ο ήχος θα τοποθετηθεί στο σημείο (-2,-4,-4) και στην συνεχεία θα εκτελεστεί. Αντίστοιχα αν ο ήχος δεν είναι σε κατάσταση AL_PLAYING και το flag Left είναι ψευδές τότε ο ήχος θα τοποθετηθεί στο σημείο (2,-4,-4) και στην συνεχεία θα εκτελεστεί. Πρέπει να σημειωθεί ότι, επειδή ο ήχος είναι μικρής διάρκειας και για να μην δημιουργεί δυσάρεστη αίσθηση, το συνεχές Panning έχει τοποθετηθεί αρκετά σιγά και φαίνεται σαν κομμάτι του δεύτερου ήχου, συμπληρώνοντας το ηχητικό περιβάλλον.

Ο τέταρτος και τελευταίος ήχος είναι τοποθετημένος δεξιά–πάνω–μέσα (1, 4,-2). Σε σχέση με τους υπόλοιπους ήχους που έχουν τοποθετηθεί αρκετά μέσα (-4) παρουσιάζεται σχετικά κοντά(-2). Ο συγκεκριμένος ήχος παρόλο που δεν είναι σε Loop ON, ουσιαστικά εκτελεί μια Loop διαδικασία. Ο λόγος που δεν τέθηκε σε Loop ON είναι γιατί θα δημιουργούσε μια δυσάρεστη αίσθηση συνέχειας. Αντ' αυτού χρησιμοποιήθηκε ένας «πρόχειρος» μετρητής, που όταν φτάσει σε ένα συγκεκριμένο αριθμό μηδενίζεται και θέτει ένα flag time αληθές. Όταν το flag time είναι αληθές ο ήχος μπορεί να εκτελεστεί.

```
/*
 *                OpenAL SDK 1.1
 *           Example Code: Multiple Sources
 *
 *    1.Initializing & exiting OpenAL and Alut
 *    2.Print red cooler error code
 *    3.Load a .wav Queue buffer
 *    4.Load a .wav  User buffer
 *    5.Load a .wav  environment buffer
 *    6.Use alSourceQueueBuffers for attach buffer
 *    7.Generate static source and attach buffer for all source
 *    8.Set default Listener Values
 *    8.User Interface For Queue sound Gain and user sound
 *    9.Multi Position Sound LR panning (down environment)
 *   10.Set Sound Position Right UP
 *   11.Clean up by deleting Source and Buffer
*/
```

/*Include.h*/

```
#ifndef INCLUDE_H_
#define INCLUDE_H_

/*Get some classic includes*/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <Windows.h>
/*This include work ONLY MinGW. NOT compiler work Cygwin...(?) */
#include <conio.h>

/*includes FULL OpenAL*/
#include <al.h>
#include <alc.h>
#include <alut.h>
#include <efx.h>
#include <efx-creative.h>
#include <xram.h>


/************** GENERAL FUNCTION **************/
void whitePrint(const char *Text);/*Print White Text*/
void RedPrint(const char *Text);/*Print Red  Text */
/*Print Red AL-ALC Error code & Text */
void RedALErrorPrint(const char *Text, ALenum Error);
/*Print Red ALUT Error code & Text */
void RedALUTErrorPrint(const char *Text, ALenum Error);

void ExitEnterPress(void); /*quit programme*/
void StartPrint();/*Start program print*/

/****** INITIALIZATION & EXIT FUNCTION ******/
ALboolean InitOpenAL();/*Initialization OpenAL*/
ALboolean CloseOpenAL();/*Exit-Close OpenAL*/
ALboolean InitALUT();/*Initialization ALUT*/
ALboolean CloseALUT ();/*Exit-Close ALUT*/
#endif /*INCLUDE_H_*/
```

/*Include.c*/

```
/*Include Function File*/

#include "include.h"
/************** GENERAL FUNCTION **************/

void StartPrint(){/*Start program print*/

HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
SetConsoleTextAttribute(h, FOREGROUND_RED | FOREGROUND_GREEN |
                        FOREGROUND_BLUE |FOREGROUND_INTENSITY);

printf ( "*************************************************\n" );
printf ( "****************** OpenAL SDK 1.1 ******************\n" );
printf ( "********* Example Code: 5.Multiple Sources *********\n" );
printf ( "*************************************************\n" );
      return ;
}
```

```
/* Colour info:
 * Win32 Console Applications Tutorials- Part 4 colour.
 * http://www.adrianxw.dk/SoftwareSite/Consoles/Consoles4.html*/

void whitePrint(const char *Text) /*Print White Text*/
{
HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
SetConsoleTextAttribute(h, FOREGROUND_RED | FOREGROUND_GREEN |
                        FOREGROUND_BLUE |FOREGROUND_INTENSITY);
printf("%s", Text);
}

void RedPrint(const char *Text) /*Print Red  Text */
{
HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
SetConsoleTextAttribute(h, FOREGROUND_RED |FOREGROUND_INTENSITY);
printf("%s", Text);
}

/*Print Red AL-ALC Error code & Text */
void RedALErrorPrint(const char *Text, ALenum Error)
{
HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );

SetConsoleTextAttribute(h, FOREGROUND_RED |FOREGROUND_INTENSITY);
printf("%s%s\n", Text, alGetString(Error));
}

/*Print Red ALUT Error code & Text */
void RedALUTErrorPrint(const char *Text, ALenum Error)
{
HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );

SetConsoleTextAttribute(h, FOREGROUND_RED |FOREGROUND_INTENSITY);
      printf("%s%s\n",Text, alutGetErrorString(Error));
}
```

```
void ExitEnterPress(void) /*quit programme*/
{
      whitePrint ( "\nPress [Enter] to quit . . ." );
      fflush ( stdout );
      getchar();
      exit(0); /*Returns 0 to the operating system*/
}
```

```
/****** INITIALIZATION & EXIT FUNCTION ******/

ALboolean InitOpenAL()/*Initialization OpenAL manually*/
{

ALCcontext        *pContext = NULL;
ALCdevice         *pDevice = NULL;
ALboolean          bReturn = AL_FALSE;
ALenum            Error; /* save error status*/
ALCboolean        currentcon ;/*test for alcMakeContextCurrent*/
```

```
pDevice = alcOpenDevice(NULL); /*Open default device*/
if (pDevice){ /*pDevice != NULL*/
      alcGetError(pDevice);/*Clear error code*/
      /*creates a context using a default device*/
      pContext = alcCreateContext(pDevice, NULL);
      if (pContext){/*pContext != NULL*/
     /*set active context-Makes a pContext the current context*/
            currentcon = alcMakeContextCurrent(pContext);
            if (currentcon == ALC_TRUE){
                  bReturn = AL_TRUE;
            }
            else{/*alcMakeContextCurrent(pContext)!= ALC_TRUE */
                  Error = alcGetError(pDevice);
                  RedALErrorPrint( "Unable to make OpenAL context
                                    current: !!!Error!!!", Error );
            }
      }
      else {/*pContext = NULL*/
            Error = alcGetError(pDevice);
            RedALErrorPrint ( "Unable to create an OpenAL context:
                              !!!Error!!!", Error);
      }
}
else{/*pDevice = NULL*/

      HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
      SetConsoleTextAttribute(h, FOREGROUND_RED
                              |FOREGROUND_INTENSITY);
      printf("Unable to Open default device :!!!Error!!!");

}
return bReturn;
}
```

```
ALboolean CloseOpenAL()/*Exit-Close OpenAL*/
{
ALCcontext *pContext;
ALCdevice *pDevice;
ALCboolean currentcon ;/*test for alcMakeContextCurrent*/

pContext = alcGetCurrentContext(); /*open the current context */
if (alcGetCurrentContext() == NULL){
      whitePrint ( "\n WARNING: There is no current context, NULL is
                  returned........" );
}
/*open a context's device pointer*/
pDevice = alcGetContextsDevice(pContext);
/*makes NULL context the current context*/
currentcon = alcMakeContextCurrent(NULL);
if (currentcon != ALC_TRUE){
      whitePrint ( "WARNING: failure to makes NULL context the
                    current context......" );
}
alcDestroyContext(pContext);/*destroys a context*/
alcCloseDevice(pDevice);/*closes a device*/
return AL_TRUE;
}
```

```
ALboolean  InitALUT()/*Initialization ALUT*/
{
      ALboolean   bReturn = AL_FALSE;
      ALenum      Error; /*Save ALUT error status*/

      alutGetError();/*Clear error code*/

      if (!alutInitWithoutContext( NULL , NULL )) {
            Error = alutGetError();
            RedALUTErrorPrint("!!!Error!!!", Error);
      }
      else
      {
      bReturn = AL_TRUE;
      }
      return bReturn;
}
```

```
ALboolean CloseALUT()/*Exit-Close ALUT */
{
      ALboolean   bReturn = AL_FALSE;
      ALenum      Error; /*Save ALUT error status*/

      alutGetError();/*Clear error code*/

      if (!alutExit()) {
            Error = alutGetError();
            RedALUTErrorPrint("Close ALUT : !!!Error!!!", Error);
      }
      bReturn = AL_TRUE;
      return bReturn;
}
```

/* Main.c*/

```
/*Include Function File*/

#include "include.h"


BOOL  done=FALSE;       // Bool Variable To Exit Loop - Windows
#define NUM_Q_BUFFERS 8 //Queue Buffer

int main ( void ){

/****************** Definition Variable *****************/

ALboolean   AReturn = AL_FALSE; // TRUE OR FALSE Function  Return
ALboolean   BReturn = AL_FALSE; // TRUE OR FALSE Function  Return

ALbyte      keys;               //  Keyboard  Keys
ALenum      Error;              // Save AL and ALUT error status
ALfloat     UsGain = 0.5;       // User Gain

ALuint      QueueBuffers[NUM_Q_BUFFERS];  // Queue Buffer Name
ALuint      QueueSources;                 // Queue source Name

ALuint      warningB;         // warning Buffer Name
ALuint      warningS;         // warning source Name
```

```
ALuint      UserSoundB; // User Buffer Name
ALuint      UserSoundS; // User source Name

// User Default source Position
ALfloat     SourceUserSoundPos[] = { 0.0, 0.0, 0.0 };

ALuint      EnvBackB;   // environment back Buffer Name
ALuint      EnvBackS;   // environment back source Name

//environment back source Position (z= -1)
ALfloat     SourceEnvBackPos[] = { 0.0, 0.0,-1.0 };

ALuint      EnvLRDownB; // environment LR Down Buffer Name
ALuint      EnvLRDownS; // environment LR Down source Name
ALint       status;     //Source State
ALint       Left = AL_TRUE;//flag Source State set left position

//environment Left  Down source Position
ALfloat     SourceLDownPos[] = { -2.0, -4.0,-4.0 };
//environment Right Down source Position
ALfloat     SourceRDownPos[] = {  2.0, -4.0,-4.0 };

ALuint      EnvRUPB; // environment Right UP Buffer Name
ALuint      EnvRUPS; // environment Right UP source Name
ALint       status2; // Source State
ALint       Time = AL_TRUE;//flag Source timer State
ALint       i=1;     // timer

//environment Right UP source Position
ALfloat     SourceRUPos[] = {  1.0, 4.0,-2.0 };

// Position of the listener.
ALfloat ListenerPos[] = { 0.0, 0.0, 0.0 };
// Orientation of the listener. (first 3 elements are "at",
// second 3 are "up")
ALfloat ListenerOri[] = { 0.0, 0.0, -1.0, 0.0, 1.0, 0.0 };


//Sound Path :..../sounds/

//Background Sound name
const char  *FileName1  = "sounds\\tantra_intro.wav";
const char  *FileName2  = "sounds\\Mandara1stField2.wav";
const char  *FileName3  = "sounds\\Japan_Battle_2nd.wav";
const char  *FileName4  = "sounds\\Japan_Battle_Final.wav";
const char  *FileName5  = "sounds\\India_Battle_3rd.wav";
const char  *FileName6  = "sounds\\Japan_Field_2nd.wav";
const char  *FileName7  = "sounds\\mudha.wav";
const char  *FileName8  = "sounds\\TrimuritiBattleField.wav";
//warning Sound name
const char  *WarSound   = "sounds\\ding.wav";
//User Sound name
const char  *UserSound  = "sounds\\Gun.wav";
//environment Sound name
const char  *EnvBack    = "sounds\\planet_horror.wav";
const char  *EnvLRDown  = "sounds\\R1FX004.WAV";
const char  *EnvRUP     = "sounds\\R1FX013.WAV";
```

```
/************ INITIALIZATION OpenAL AND ALUT ************/

StartPrint(); //Start program print
AReturn   = InitOpenAL();
BReturn   = InitALUT();
if ( AReturn == AL_TRUE && BReturn == AL_TRUE ){
      whitePrint( "\n Initialization OpenAL Complete:\n\n" );
}
else {
      RedPrint("\n !ERROR! Initialization OpenAL NOT Complete\n\n");
      CloseALUT();
      CloseOpenAL();
      ExitEnterPress();
}
```

```
/****************** MAIN PROGRAM LOAD ******************/

alutGetError();/*Clear ALUT error code*/
//Load a .WAV Queue sound
whitePrint( "\n Load a .WAV Queue sound :\n" );
QueueBuffers[0]=alutCreateBufferFromFile ( FileName1 );
if (QueueBuffers[0] == AL_NONE ){
      Error = alutGetError();
      RedALUTErrorPrint("\n!!! Error!!! loading file:" ,Error);
      printf("Failed to load %s\n", FileName1);
      alDeleteBuffers(1,&QueueBuffers[0]);//delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
else{
      printf("%s\n", FileName1);
}
QueueBuffers[1]=alutCreateBufferFromFile ( FileName2 );
if (QueueBuffers[1] == AL_NONE ){
      Error = alutGetError();
      RedALUTErrorPrint("\n !!!Error!!! loading file: ",Error);
      printf("Failed to load %s\n", FileName2);
      alDeleteBuffers(1,&QueueBuffers[1]);//delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
else{
      printf("%s\n", FileName2);
}
QueueBuffers[2]=alutCreateBufferFromFile ( FileName3 );
if (QueueBuffers[2] == AL_NONE ){
      Error = alutGetError();
      RedALUTErrorPrint(" \n !!!Error!!! loading file: ",Error);
      printf("Failed to load %s\n", FileName3);
      alDeleteBuffers(1,&QueueBuffers[2]);//delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
else{
      printf("%s\n", FileName3);
}
```

```
QueueBuffers[3]=alutCreateBufferFromFile ( FileName4 );
if (QueueBuffers[3] == AL_NONE ){
      Error = alutGetError();
      RedALUTErrorPrint("\n !!!Error!!! loading file: ",Error);
      printf("Failed to load %s\n", FileName4);
      alDeleteBuffers(1,&QueueBuffers[3]);//delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL//
      ExitEnterPress();//quit programme
}
else{
      printf("%s\n", FileName4);
}


QueueBuffers[4]=alutCreateBufferFromFile ( FileName5 );
if (QueueBuffers[4] == AL_NONE ){
      Error = alutGetError();
      RedALUTErrorPrint("\n !!!Error!!! loading file: ",Error);
      printf("Failed to load %s\n", FileName5);
      alDeleteBuffers(1,&QueueBuffers[4]);//delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
else{
      printf("%s\n", FileName5);
}

QueueBuffers[5]=alutCreateBufferFromFile ( FileName6 );
if (QueueBuffers[5] == AL_NONE ){
      Error = alutGetError();
      RedALUTErrorPrint("\n !!!Error!!! loading file: ",Error);
      printf("Failed to load %s\n", FileName6);
      alDeleteBuffers(1,&QueueBuffers[5]);//delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
else{
      printf("%s\n", FileName6);
}


QueueBuffers[6]=alutCreateBufferFromFile ( FileName7 );
if (QueueBuffers[6] == AL_NONE ){
      Error = alutGetError();
      RedALUTErrorPrint("\n !!!Error!!! loading file: ",Error);
      printf("Failed to load %s\n", FileName7);
      alDeleteBuffers(1,&QueueBuffers[6]);//delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
else{
      printf("%s\n", FileName7);
}
```

```
QueueBuffers[7]=alutCreateBufferFromFile ( FileName8 );
if (QueueBuffers[7] == AL_NONE ){
	Error = alutGetError();
	RedALUTErrorPrint("\n !!!Error!!! loading file: ",Error);
	printf("Failed to load %s\n", FileName8);
	alDeleteBuffers(1,&QueueBuffers[7]);//delete Buffer
	CloseALUT();//Exit-Close ALUT
	CloseOpenAL();//Exit-Close OpenAL
	ExitEnterPress();//quit programme
}
else{
	printf("%s\n", FileName8);
}

//Load a .WAV user sound
whitePrint( "\n\n Load a .WAV User sound :\n" );

warningB=alutCreateBufferFromFile ( WarSound );
if (warningB == AL_NONE ){
	Error = alutGetError();
	RedALUTErrorPrint("\n !!!Error!!! loading file: ",Error);
	printf("Failed to load %s\n", WarSound);
	alDeleteBuffers(1,&warningB);//delete Buffer
	CloseALUT();//Exit-Close ALUT
	CloseOpenAL();//Exit-Close OpenAL
	ExitEnterPress();//quit programme
}
else{
	printf("%s\n", WarSound);
}
UserSoundB=alutCreateBufferFromFile ( UserSound );
if (UserSoundB == AL_NONE ){
	Error = alutGetError();
	RedALUTErrorPrint("\n !!!Error!!! loading file: ",Error);
	printf("Failed to load %s\n", UserSound);
	alDeleteBuffers(1,&UserSoundB);//delete Buffer
	CloseALUT();//Exit-Close ALUT
	CloseOpenAL();//Exit-Close OpenAL
	ExitEnterPress();//quit programme
}
else{
	printf("%s\n", UserSound);
}

//Load a .WAV environment sound
whitePrint( "\n\n Load a .WAV Environment sound :\n" );
EnvBackB=alutCreateBufferFromFile ( EnvBack );
if (EnvBackB == AL_NONE ){
	Error = alutGetError();
	RedALUTErrorPrint("\n !!!Error!!! loading file: ",Error);
	printf("Failed to load %s\n", EnvBack);
	alDeleteBuffers(1,&EnvBackB);//delete Buffer
	CloseALUT();//Exit-Close ALUT
	CloseOpenAL();//Exit-Close OpenAL
	ExitEnterPress();//quit programme
}
else{
	printf("%s\n", EnvBack);
}
```

```
EnvLRDownB=alutCreateBufferFromFile ( EnvLRDown );
if (EnvLRDownB == AL_NONE ){
      Error = alutGetError();
      RedALUTErrorPrint("\n !!!Error!!! loading file:" ,Error);
      printf("Failed to load %s\n", EnvLRDown);
      alDeleteBuffers(1,&EnvLRDownB);//delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
else{
      printf("%s\n", EnvLRDown);
}


EnvRUPB=alutCreateBufferFromFile ( EnvRUP );
if (EnvRUPB == AL_NONE ){
      Error = alutGetError();
      RedALUTErrorPrint("\n !!!Error!!! loading file: ",Error);
      printf("Failed to load %s\n", EnvRUP);
      alDeleteBuffers(1,&EnvLRDownB);//delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
else{
      printf("%s\n", EnvRUP);
}
```

```
/****************** MAIN PROGRAM PLAY ******************/

alGetError();      //Clear AL error code
//QUEUE SOUND
alGenSources (1, &QueueSources);//Generate a single source for queue
Error = alGetError ();
if ( Error != AL_NO_ERROR )
{
      RedALErrorPrint("\n !!!Error!!! Failure to Generate a Source:",Error);
      alDeleteSources(1, &QueueSources);//delete source
      alDeleteBuffers(8,QueueBuffers); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}

// Queue Sound Data
alSourceQueueBuffers ( QueueSources, NUM_Q_BUFFERS, QueueBuffers);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
{
      RedALErrorPrint("\n !!!Error!!! Failure to Attach Source to Queue
                  Buffer:", Error);
      alDeleteSources(1, &QueueSources);//delete source
      alDeleteBuffers(8,QueueBuffers); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
```

```
//WARNING SOUND
alGenSources (1, &warningS);//Generate a single source for warning
Error = alGetError ();
if ( Error != AL_NO_ERROR )
{
      RedALErrorPrint("\n !!!Error!!! Failure to Generate a Source:",Error);
      alDeleteSources(1, &warningS);//delete source
      alDeleteBuffers(1,&warningB); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}

//Attach warning Source to Buffer
alSourcei (warningS, AL_BUFFER, warningB);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
{
      RedALErrorPrint("\n !!!Error!!! Failure to Attach Source to
                       Buffer:", Error);
      alDeleteSources(1, &warningS);//delete source
      alDeleteBuffers(1,&warningB); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}


//USER SOUND
alGenSources (1, &UserSoundS);//Generate a single source for user
Error = alGetError ();
if ( Error != AL_NO_ERROR )
{
      RedALErrorPrint("\n !!!Error!!! Failure to Generate a Source:",Error);
      alDeleteSources(1, &UserSoundS);//delete source
      alDeleteBuffers(1,&UserSoundB); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}


//Attach warning Source to Buffer
alSourcei (UserSoundS, AL_BUFFER, UserSoundB);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
{
      RedALErrorPrint("\n !!!Error!!! Failure to Attach Source to
                     Buffer:", Error);
      alDeleteSources(1, &UserSoundS);//delete source
      alDeleteBuffers(1,&UserSoundB); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
```

```
// ENVIRONMENT SOUND
alGenSources (1, &EnvBackS);//Generate a single source
Error = alGetError ();
if ( Error != AL_NO_ERROR )
{
      RedALErrorPrint("\n !!!Error!!! Failure to Generate a Source:",Error);
      alDeleteSources(1, &EnvBackS);//delete source
      alDeleteBuffers(1,&EnvBackB); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
//Attach  Source to Buffer
alSourcei (EnvBackS, AL_BUFFER, EnvBackB);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
{
      RedALErrorPrint("\n !!!Error!!! Failure to Attach Source to
                  Buffer: ",Error);
      alDeleteSources(1, &EnvBackS);//delete source
      alDeleteBuffers(1,&EnvBackB); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
alGenSources (1, &EnvLRDownS);//Generate a single source
Error = alGetError ();
if ( Error != AL_NO_ERROR )
{
      RedALErrorPrint("\n !!!Error!!! Failure to Generate a Source:",Error);
      alDeleteSources(1, &EnvLRDownS);//delete source
      alDeleteBuffers(1,&EnvLRDownB); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
//Attach Source to Buffer
alSourcei (EnvLRDownS, AL_BUFFER, EnvLRDownB);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
{
      RedALErrorPrint("\n !!!Error!!! Failure to Attach Source to
                  Buffer: ",Error);
      alDeleteSources(1, &EnvLRDownS);//delete source
      alDeleteBuffers(1,&EnvLRDownB); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
alGenSources (1, &EnvRUPS);//Generate a single source
Error = alGetError ();
if ( Error != AL_NO_ERROR )
{
      RedALErrorPrint("\n !!!Error!!! Failure to Generate a Source:",Error);
      alDeleteSources(1,&EnvRUPS);//delete source
      alDeleteBuffers(1,&EnvRUPB); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}
```

```
//Attach Source to Buffer
alSourcei (EnvRUPS, AL_BUFFER, EnvRUPB);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
{
      RedALErrorPrint("\n !!!Error!!! Failure to Attach Source to
                       Buffer: ",Error);
      alDeleteSources(1,&EnvRUPS);//delete source
      alDeleteBuffers(1,&EnvRUPB); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}


//Set Listener Values
alListenerfv(AL_POSITION,    ListenerPos);
alListenerfv(AL_ORIENTATION, ListenerOri);

//Star Queue  Sound
alSourcef (QueueSources, AL_GAIN, UsGain);//GAIN SET 0.5
alSourcei(QueueSources, AL_LOOPING, AL_TRUE);//Loop ON
alSourcePlay (QueueSources);//play a source

//Star environment  Sound

//Set Sound position
alSourcefv(EnvBackS, AL_POSITION, SourceEnvBackPos);
alSourcef (EnvBackS, AL_GAIN, 0.4);//Set GAIN
alSourcei(EnvBackS, AL_LOOPING, AL_TRUE);//Loop ON
alSourcePlay (EnvBackS);//play a source


whitePrint("\n\n\n\n");
whitePrint("Select from the following options:\n\n");
whitePrint("Press [Up Arrow]   to Increase Background Gain \n");
whitePrint("Press [Down Arrow] to Decrease Background Gain \n");
whitePrint("Press [Spacebar]   to Start your Sound\n");
whitePrint("Press [Enter]      to quit\n\n\n");

whitePrint("Info: If the gain of the background sound has reached the
           Maximum\n");
whitePrint(" or Minimum level you will listen a warning sound\n\n");
whitePrint(" Press [BackSpace] to listen to the warning sound\n");
whitePrint("\n\n\n\n\n\n\n\n\n\n");


do
{      // Get Source State
      alGetSourcei (EnvLRDownS, AL_SOURCE_STATE, &status);
      if (status != AL_PLAYING){ // if source not play
            if (Left ==AL_TRUE){ // if left flag is true
                  //Set left Sound position
                  alSourcefv(EnvLRDownS, AL_POSITION,
                              SourceLDownPos);
                  alSourcef (EnvLRDownS, AL_GAIN, 0.4);//Set GAIN
                  Left =AL_FALSE;// set flag False
                  alSourcePlay (EnvLRDownS);//play user source sound
            }
```

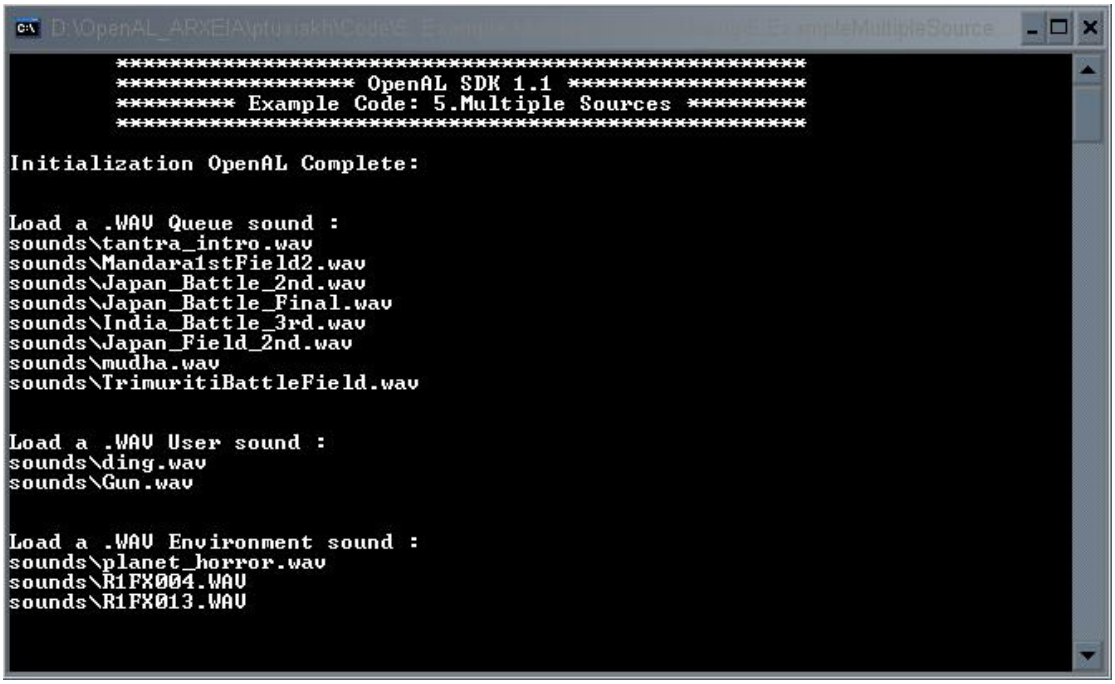
```
        else{
             //Set Right Sound position
              alSourcefv(EnvLRDownS, AL_POSITION,
                        SourceRDownPos);
              alSourcef (EnvLRDownS, AL_GAIN, 0.4); //Set GAIN
              Left =AL_TRUE;// set left flag true
              alSourcePlay (EnvLRDownS); //play user source sound

        }
}


// Get Source State
alGetSourcei (EnvRUPS, AL_SOURCE_STATE, &status2);
if (status2 != AL_PLAYING){//if source not play

        if (Time == AL_TRUE){ //if timer flag is true
              //Set Sound position
              alSourcefv(EnvRUPS, AL_POSITION, SourceRUPos);
              alSourcef (EnvLRDownS, AL_GAIN, 0.4); //Set GAIN
              Time =AL_FALSE; // set flag False
              alSourcePlay (EnvRUPS); //play user source sound

        }
        else{
              i+=1;

              if(i==100000){
                    Time = AL_TRUE;// set flag True
                    i=0;
              }
        }
}


if(kbhit()){//if keys pressed

    keys= getch();//Get keys

      if (keys == 13){        //If Enter Being Pressed
            done=TRUE;        //Exit Loop - Windows
      }

      if (keys == 72){        // If Up Arrow Being Pressed
            UsGain+=0.05f;    // If So, Increase UsGain

            if (UsGain >= 1){ // If usGain >= 1
                  UsGain =1;  // set usGain = 1
                  alSourcePlay (warningS);//play warning sound
            }

            alSourcef (QueueSources, AL_GAIN, UsGain);
      }
```

```
            if (keys == 80 ){       // If Down Arrow Being Pressed
                  UsGain-=0.05f;    // Decrease UsGain

                  if (UsGain <= 0 ){ // If usGain <= 0
                        UsGain =0;   // set usGain = 0
                        alSourcePlay (warningS);//play warning sound
                  }
                  alSourcef (QueueSources, AL_GAIN, UsGain);
            }

            if (keys == 8){// If BackSpace Being Pressed
                  alSourcePlay (warningS);//play warning sound
            }

            if (keys == 32){ //If Spacebar Being Pressed
                  alSourcefv(UserSoundS, AL_POSITION,
                              SourceUserSoundPos);//Set Sound position
                  alSourcePlay (UserSoundS);//play user source sound
            }
      }

}while (!done);
```

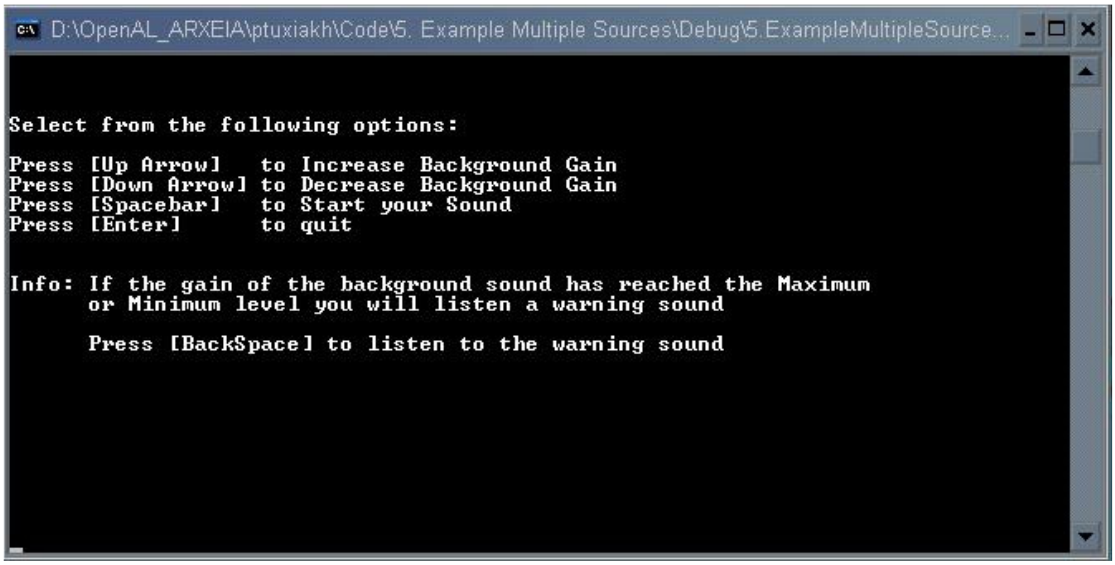
```
/****************** OUT PROGRAMME ********************/
alDeleteSources(1,&EnvRUPS);//delete environment Right Up source
alDeleteBuffers(1,&EnvRUPB); //delete environment Right Up Buffer

//delete environment Left Right Down source
alDeleteSources(1,&EnvLRDownS);
//delete environment Left Right Down Buffer
alDeleteBuffers(1,&EnvLRDownB);

alDeleteSources(1, &EnvBackS);//delete environment Back source
alDeleteBuffers(1,&EnvBackB); //delete environment Back Buffer

alDeleteSources(1, &UserSoundS);//delete User source
alDeleteBuffers(1,&UserSoundB); //delete User Buffer

alDeleteSources(1, &warningS);//delete warning source
alDeleteBuffers(1,&warningB); //delete warning Buffer

alDeleteSources(1, &QueueSources);//delete Queue source
alDeleteBuffers(8,QueueBuffers);//Delete Queue Buffer

CloseALUT();//Exit-Close ALUT
CloseOpenAL();//Exit-Close OpenAL
return 0;
}
```

Στο συγκεκριμένο παράδειγμα χρησιμοποιήθηκαν διάφοροι τρόποι ώστε να μην δημιουργηθεί η αίσθηση της απόλυτης συνέχειας. Σε μια τρισδιάστατη εφαρμογή, οι ήχοι εκτελούνται όταν συμβεί μια συγκεκριμένη ενέργεια από τον χρήστη ή όταν ένας εσωτερικός system timer δώσει εντολή ότι «είναι ώρα» ένα ήχος να εκτελεστεί. Αποτέλεσμα αυτού είναι η αίσθηση της συνέχειας να μην είναι καθόλου εμφανής υποβοηθούμενη βέβαια πάντα από την εικόνα.

Όπως έχει προαναφερθεί αρχικά φορτώνονται τα αρχεία ήχου :



Στην συνέχεια ο χρήστης μπορεί να ελέγξει ορισμένα χαρακτηριστικά του περιβάλλοντος:



6. Example: System info - Extensions

Στο παράδειγμα αυτό ο χρήστης έχει την δυνατότητα να δει τα βασικά Extension του συστήματος OpenAL. Ταυτόχρονα εξετάζεται και η δυνατότητα του συστήματος του χρήστη για την εκτέλεση λειτουργιών μέσω της EFX.

Μετά το Initialize εκτυπώνεται η έκδοση της OpenAL που χρησιμοποιείται, καθώς και πληροφορίες για τα διαθέσιμα Device. Στην συνέχεια εκτυπώνονται οι διαθέσιμες Context Extensions (είναι πάντα 3-ALC_ENUMERATION_EXT, ALC_EXT_CAPTURE, ALC_EXT_EFX) και ελέγχεται αν είναι διαθέσιμες από το σύστημα. Ακολουθεί η εκτύπωση των διαθέσιμων AL Extensions (είναι πάντα 9-AL_EXT_OFFSET, AL_EXT_LINEAR_DISTANCE, AL_EXT_EXPONENT_DISTANCE, EAX, EAX2.0, EAX3.0, EAX4.0, EAX3.0EMULATED, EAX4.0EMULATED), ενώ γίνεται και έλεγχος για διαθέσιμη X-RAM στο σύστημα (ειδική μνήμη που ορισμένες κάρτες ήχου διαθέτουν).

Στην συνέχεια γίνονται Setup οι EFX Functions και ακολουθούν δοκιμές για την υποστήριξη της EFX από την κάρτα ήχου. Αρχικά ζητείται από το σύστημα να δημιουργήσει έναν αρκετά μεγάλο αριθμό Auxiliary Effect Slots (120) και μόλις δημιουργηθεί ένα error η λειτουργία σταματάει και εκτυπώνει τον αριθμό των Auxiliary Effect Slots, που μπορεί να υποστηρίξει η κάρτα ήχου. Στην συνέχεια γίνεται ερώτηση (ALC_MAX_AUXILIARY_SENDS) για τον μέγιστο αριθμό Auxiliary Effect Slots, που μπορεί να διατεθούν ανά Source. Τέλος εξετάζονται αν όλα τα Effects και Filters της EFX είναι υποστηρίξιμα από την κάρτα ήχου.

```
/*****************************************************************
 *                            OpenAL SDK 1.1
 *                             Example Code
 *
 *          1.Initializing & exiting OpenAL
 *          2.OpenAL Info: AL_VENDOR, AL_VERSION, AL_RENDERER
 *          3.Device info: ALC_DEVICE_SPECIFIER,
 *          ALC_DEFAULT_DEVICE_SPECIFIER, ALC_CAPTURE_DEVICE_SPECIFIER,
 *          ALC_CAPTURE_DEFAULT_DEVICE_SPECIFIER
 *
 *          4.ALC Extensions: ALC_ENUMERATION_EXT, ALC_EXT_CAPTURE,
 *                            ALC_EXT_EFX
 *          5.Query for ALC Extensions Support
 *
 *          6.AL Extensions: AL_EXT_OFFSET, AL_EXT_LINEAR_DISTANCE,
 *          AL_EXT_EXPONENT_DISTANCE, EAX, EAX2.0, EAX3.0,EAX4.0,
 *          EAX3.0EMULATED, EAX4.0EMULATED
 *
 *          7.Query for AL Extensions Support
 *
 *          8.Query for EAX-RAM Support: "EAX-RAM"
 *
 *          9.EFX Function setup
 *         10.Try to create 120 Auxiliary Effect Slots
 *
 *         11.Test Max support Auxiliary Effect Slots and
 *            Auxiliary Sends per Source
 *         12.Test Device Support Available Effects and Filter
*****************************************************************/
```

```
/*Get some classic includes*/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <Windows.h>

/*includes FULL OpenAL*/
#include <al.h>
#include <alc.h>
#include <alut.h>
#include <efx.h>
#include <xram.h>


// Effect objects
LPALGENEFFECTS alGenEffects = NULL;
LPALDELETEEFFECTS alDeleteEffects = NULL;
LPALISEFFECT alIsEffect = NULL;
LPALEFFECTI alEffecti = NULL;
LPALEFFECTIV alEffectiv = NULL;
LPALEFFECTF alEffectf = NULL;
LPALEFFECTFV alEffectfv = NULL;
LPALGETEFFECTI alGetEffecti = NULL;
LPALGETEFFECTIV alGetEffectiv = NULL;
LPALGETEFFECTF alGetEffectf = NULL;
LPALGETEFFECTFV alGetEffectfv = NULL;


//Filter objects
LPALGENFILTERS alGenFilters = NULL;
LPALDELETEFILTERS alDeleteFilters = NULL;
LPALISFILTER alIsFilter = NULL;
LPALFILTERI alFilteri = NULL;
LPALFILTERIV alFilteriv = NULL;
LPALFILTERF alFilterf = NULL;
LPALFILTERFV alFilterfv = NULL;
LPALGETFILTERI alGetFilteri = NULL;
LPALGETFILTERIV alGetFilteriv = NULL;
LPALGETFILTERF alGetFilterf = NULL;
LPALGETFILTERFV alGetFilterfv = NULL;


// Auxiliary slot object
LPALGENAUXILIARYEFFECTSLOTS alGenAuxiliaryEffectSlots = NULL;
LPALDELETEAUXILIARYEFFECTSLOTS alDeleteAuxiliaryEffectSlots = NULL;
LPALISAUXILIARYEFFECTSLOT alIsAuxiliaryEffectSlot = NULL;
LPALAUXILIARYEFFECTSLOTI alAuxiliaryEffectSloti = NULL;
LPALAUXILIARYEFFECTSLOTIV alAuxiliaryEffectSlotiv = NULL;
LPALAUXILIARYEFFECTSLOTF alAuxiliaryEffectSlotf = NULL;
LPALAUXILIARYEFFECTSLOTFV alAuxiliaryEffectSlotfv = NULL;
LPALGETAUXILIARYEFFECTSLOTI alGetAuxiliaryEffectSloti = NULL;
LPALGETAUXILIARYEFFECTSLOTIV alGetAuxiliaryEffectSlotiv = NULL;
LPALGETAUXILIARYEFFECTSLOTF alGetAuxiliaryEffectSlotf = NULL;
LPALGETAUXILIARYEFFECTSLOTFV alGetAuxiliaryEffectSlotfv = NULL;
```

```
/*EFX Function setup*/
ALboolean EFXFunctionSetup(){

ALCdevice *pDevice = NULL;
ALCcontext *pContext = NULL;
ALboolean   Setup = AL_FALSE;
pContext = alcGetCurrentContext();
pDevice = alcGetContextsDevice(pContext);

if (alcIsExtensionPresent(pDevice, (ALCchar*)ALC_EXT_EFX_NAME)){

// Get function pointers
alGenEffects = (LPALGENEFFECTS)alGetProcAddress("alGenEffects");
alDeleteEffects = (LPALDELETEEFFECTS )
                    alGetProcAddress("alDeleteEffects");
alIsEffect = (LPALISEFFECT )alGetProcAddress("alIsEffect");
alEffecti = (LPALEFFECTI)alGetProcAddress("alEffecti");
alEffectiv = (LPALEFFECTIV)alGetProcAddress("alEffectiv");
alEffectf = (LPALEFFECTF)alGetProcAddress("alEffectf");
alEffectfv = (LPALEFFECTFV)alGetProcAddress("alEffectfv");
alGetEffecti = (LPALGETEFFECTI)alGetProcAddress("alGetEffecti");
alGetEffectiv = (LPALGETEFFECTIV)alGetProcAddress("alGetEffectiv");
alGetEffectf = (LPALGETEFFECTF)alGetProcAddress("alGetEffectf");
alGetEffectfv = (LPALGETEFFECTFV)alGetProcAddress("alGetEffectfv");

alGenFilters = (LPALGENFILTERS)alGetProcAddress("alGenFilters");
alDeleteFilters =
(LPALDELETEFILTERS)alGetProcAddress("alDeleteFilters");
alIsFilter = (LPALISFILTER)alGetProcAddress("alIsFilter");
alFilteri = (LPALFILTERI)alGetProcAddress("alFilteri");
alFilteriv = (LPALFILTERIV)alGetProcAddress("alFilteriv");
alFilterf = (LPALFILTERF)alGetProcAddress("alFilterf");
alFilterfv = (LPALFILTERFV)alGetProcAddress("alFilterfv");
alGetFilteri = (LPALGETFILTERI )alGetProcAddress("alGetFilteri");
alGetFilteriv= (LPALGETFILTERIV )alGetProcAddress("alGetFilteriv");
alGetFilterf = (LPALGETFILTERF )alGetProcAddress("alGetFilterf");
alGetFilterfv= (LPALGETFILTERFV )alGetProcAddress("alGetFilterfv");

alGenAuxiliaryEffectSlots =(LPALGENAUXILIARYEFFECTSLOTS)

alGetProcAddress("alGenAuxiliaryEffectSlots");
alDeleteAuxiliaryEffectSlots=(LPALDELETEAUXILIARYEFFECTSLOTS)

alGetProcAddress("alDeleteAuxiliaryEffectSlots");
alIsAuxiliaryEffectSlot=(LPALISAUXILIARYEFFECTSLOT)
                        alGetProcAddress("alIsAuxiliaryEffectSlot");
alAuxiliaryEffectSloti = (LPALAUXILIARYEFFECTSLOTI)
                        alGetProcAddress("alAuxiliaryEffectSloti");
alAuxiliaryEffectSlotiv = (LPALAUXILIARYEFFECTSLOTIV)
                         alGetProcAddress("alAuxiliaryEffectSlotiv");
alAuxiliaryEffectSlotf = (LPALAUXILIARYEFFECTSLOTF)
                        alGetProcAddress("alAuxiliaryEffectSlotf");
alAuxiliaryEffectSlotfv = (LPALAUXILIARYEFFECTSLOTFV)
                         alGetProcAddress("alAuxiliaryEffectSlotfv");
alGetAuxiliaryEffectSloti = (LPALGETAUXILIARYEFFECTSLOTI)
                       alGetProcAddress("alGetAuxiliaryEffectSloti");
```

```
alGetAuxiliaryEffectSlotiv = (LPALGETAUXILIARYEFFECTSLOTIV)
                      alGetProcAddress("alGetAuxiliaryEffectSlotiv");
alGetAuxiliaryEffectSlotf = (LPALGETAUXILIARYEFFECTSLOTF)
                       alGetProcAddress("alGetAuxiliaryEffectSlotf");
alGetAuxiliaryEffectSlotfv = (LPALGETAUXILIARYEFFECTSLOTFV)
                      alGetProcAddress("alGetAuxiliaryEffectSlotfv");

            if (alGenEffects && alDeleteEffects && alIsEffect &&
                alEffecti && alEffectiv && alEffectf && alEffectfv &&
                alGetEffecti  && alGetEffectiv && alGetEffectf &&
                alGetEffectfv && alGenFilters &&alDeleteFilters &&
                alIsFilter && alFilteri && alFilteriv && alFilterf &&
                alFilterfv && alGetFilteri && alGetFilteriv &&
                alGetFilterf && alGetFilterfv &&
                alGenAuxiliaryEffectSlots &&
                alDeleteAuxiliaryEffectSlots &&
                alIsAuxiliaryEffectSlot && alAuxiliaryEffectSloti &&
                alAuxiliaryEffectSlotiv && alAuxiliaryEffectSlotf &&
                alAuxiliaryEffectSlotfv && alGetAuxiliaryEffectSloti
                && alGetAuxiliaryEffectSlotiv &&
                alGetAuxiliaryEffectSlotf &&
                alGetAuxiliaryEffectSlotfv)
                  Setup = AL_TRUE;
      }
      return Setup;
}
```

```
int main(){

//////////////////////////////////////////////////////////
/******************* Definition Variable ******************/
ALCdevice          *pDevice = NULL;
ALCcontext         *pContext = NULL;
ALCenum             Error; //save error status
const ALchar       *szNames = NULL;//device list
ALuint              EffectSlot[120] = { 0 };//Test Auxiliary Effect
Slots
ALuint              Loop;
ALint               Sends;//Auxiliary Send per Source
ALuint              Effect;//Effect Object
ALuint              Filter;//Filter Object
```

```
//////////////////////////////////////////////////////////
/******************* Start program  ************************/
printf ( "  ************************************************\n" );
printf ( "  ****************** OpenAL SDK 1.1 ***************\n" );
printf ( "  ****** Example Code:6.System info - Extensions *****\n" );
printf ( "  ************************************************\n" );
```

```
//////////////////////////////////////////////////////////
/***************** Initializing OpenAL  *******************/

pDevice = alcOpenDevice(NULL); /*Open device*/
if (pDevice){ /*pDevice != NULL*/
      pContext = alcCreateContext(pDevice, NULL);
      if (pContext){/*pContext != NULL*/
            alcMakeContextCurrent(pContext);
      }
}
```

```
Error = alcGetError(pDevice);
if  (Error  != ALC_NO_ERROR){
      printf("\n\n Initialization OpenAL ERROR:%s", alcGetString(pDevice,
Error));
      printf ( "Press [Enter] to quit . . ." );
      fflush ( stdout );
      getchar();
      exit(0);
}
else{
      printf("\n\n Initialization OpenAL complete Successfully");
}
```

```
////////////////////////////////////////////////////////////
/********************** OpenAL Info  ***********************/
printf("\n\n OpenAL Info:\n");
/*the name of the vendor "Creative Labs Inc.".*/
printf("\n%s", alGetString(AL_VENDOR));
/*version string"1.1"*/
printf(" Openal Version %s", alGetString(AL_VERSION));
/*information about the specific rendered "Software"*/
printf(" %s", alGetString(AL_RENDERER));
```

```
////////////////////////////////////////////////////////////
/******************* OpenAL Device info  *********************/
printf("\n\n\n Device Info:\n");
//The specified (list) string for the device
szNames = alcGetString(NULL, ALC_DEVICE_SPECIFIER);
printf("\n Device List              :");

if (strlen(szNames) == 0){
      printf(" No Devices Found\n");
}
else{
      while (szNames && *szNames){
            printf(" %s", szNames);
            szNames += (strlen(szNames) + 1);
      }
}
//The specified string for the default device
printf("\n Default Device           : %s",
        alcGetString(pDevice,ALC_DEFAULT_DEVICE_SPECIFIER));

//The name of the specified capture device, or a list of all available
//capture devices if no capture device is specified.
szNames = alcGetString(NULL, ALC_CAPTURE_DEVICE_SPECIFIER);
printf("\n Capture Device List      :");
if (strlen(szNames) == 0){
      printf(" No Devices Found\n");
}
else{
      while (szNames && *szNames){
            printf(" %s", szNames);
            szNames += (strlen(szNames) + 1);
      }
}
//The name of the default capture device
printf("\n Default Capture Device   : %s",
alcGetString(pDevice,ALC_CAPTURE_DEFAULT_DEVICE_SPECIFIER));
```

```
////////////////////////////////////////////////////////////
/*************** OpenAL ALC Extensions *******************/
printf("\n\n\n Available Context Extensions:\n");

//A list of available context extensions separated by spaces.
//ALC_ENUMERATION_EXT, ALC_EXT_CAPTURE and ALC_EXT_EFX
printf("\n%s", alcGetString(pDevice, ALC_EXTENSIONS));


printf("\n\n\n System Successfully Support Context Extensions:\n");

/* Check for ALC Extensions:
 * This function queries if a specified context extension is
available.
 * ALCboolean alcIsExtensionPresent(ALCdevice *device, const ALCchar
 * extName);
 * Returns ALC_TRUE if the extension is available, ALC_FALSE if the
 * extension is not available.
 */

if (alcIsExtensionPresent(pDevice, "ALC_EXT_CAPTURE") == AL_TRUE){
      printf("\n ALC_EXT_CAPTURE ..........Yes");
}
else {
      printf("\n ALC_EXT_CAPTURE ..........No");
}

if (alcIsExtensionPresent(pDevice, "ALC_ENUMERATION_EXT") == AL_TRUE){
      printf("\n ALC_ENUMERATION_EXT .......Yes");
}
else{
      printf("\n ALC_ENUMERATION_EXT .......No");
}

if (alcIsExtensionPresent(pDevice, "ALC_EXT_EFX") == AL_TRUE){
      printf("\n ALC_EXT_EFX ..............Yes");
}
else {
      printf("\n ALC_EXT_EFX ..............No");
}
```

```
////////////////////////////////////////////////////////////
/*************** OpenAL AL Extensions  *******************/

printf("\n\n\n List Of Available Extensions:\n");
//List of available extensions separated by spaces.
//"EAX EAX2.0 EAX3.0 EAX4.0 EAX3.0EMULATED EAX4.0EMULATED"
//AND AL_EXT_OFFSET AL_EXT_LINEAR_DISTANCE AL_EXT_EXPONENT_DISTANCE

printf("\n%s", alGetString(AL_EXTENSIONS));

printf("\n\n\n System Successfully Support Available Extensions:\n");

if (alIsExtensionPresent("AL_EXT_OFFSET") == AL_TRUE){
      printf("\n AL_EXT_OFFSET.............Yes");
}
else{
      printf("\n AL_EXT_OFFSET.............No");
}
```

```
if (alIsExtensionPresent("AL_EXT_LINEAR_DISTANCE") == AL_TRUE){
      printf("\n AL_EXT_LINEAR_DISTANCE.....Yes");
}
else{
      printf("\n AL_EXT_LINEAR_DISTANCE.....No");
}
if (alIsExtensionPresent("AL_EXT_EXPONENT_DISTANCE") == AL_TRUE){
      printf("\n AL_EXT_EXPONENT_DISTANCE...Yes");
}
else{
      printf("\n AL_EXT_EXPONENT_DISTANCE...No");
}

if (alIsExtensionPresent("EAX") == AL_TRUE){
      printf("\n EAX.....................Yes");
}
else{
      printf("\n EAX.....................No");
}

if (alIsExtensionPresent("EAX2.0") == AL_TRUE){
      printf("\nEAX2.0.....................Yes");
}
else{
      printf("\nEAX2.0.....................No");
}

if (alIsExtensionPresent("EAX3.0") == AL_TRUE){
      printf("\nEAX3.0.....................Yes");
}
else{
      printf("\nEAX3.0.....................No");
}

if (alIsExtensionPresent("EAX4.0") == AL_TRUE){
      printf("\nEAX4.0.....................Yes");
}
else{
      printf("\nEAX4.0.....................No");
}

if (alIsExtensionPresent("EAX5.0") == AL_TRUE){
      printf("\nEAX5.0.....................Yes");
}
else{
      printf("\nEAX5.0.....................No");
}

if (alIsExtensionPresent("EAX3.0EMULATED ") == AL_TRUE){
      printf("\nEAX3.0EMULATED.............Yes");
}
else{
      printf("\nEAX3.0EMULATED.............No");
}

if (alIsExtensionPresent("EAX4.0EMULATED") == AL_TRUE){
      printf("\nEAX4.0EMULATED.............Yes");
}
```

```
else{
	printf("\nEAX4.0EMULATED.............No");
}

if (alIsExtensionPresent("EAX-RAM") == AL_TRUE){
	printf("\n\n EAX-RAM....................Yes");
}
else{
	printf("\n\n EAX-RAM....................No");
}
```

```
////////////////////////////////////////////////////////////
/******************* OpenAL EFX Info  ***********************/
//EFX Function setup
if (EFXFunctionSetup()){

	//Test max Auxiliary Effect Slots
	//Try to create 120 Auxiliary Effect Slots
	alGetError();//Clear AL error code
	for (Loop = 0; Loop < 120; Loop++){
		alGenAuxiliaryEffectSlots(1, &EffectSlot[Loop]);
		if (alGetError() != AL_NO_ERROR)
			break;
	}
	printf("\n\n Device supports %d Auxiliary Effect Slots\n",Loop);


	//Test max auxiliary send per source
	alcGetIntegerv(pDevice, ALC_MAX_AUXILIARY_SENDS, 1, &Sends);
	printf("Device supports %d Auxiliary Sends per Source\n",Sends);

	//Test Support Available Effects
	printf("\n Device Successfully Support Available Effects: \n");
	alGenEffects(1, &Effect);//Generate 1 Effect Object
	if (alGetError() == AL_NO_ERROR){

		alEffecti(Effect, AL_EFFECT_TYPE, AL_EFFECT_REVERB);
		if (alGetError() == AL_NO_ERROR){
			printf("\n Reverb.....................Yes");
		}
		else {
			printf("\n Reverb.....................No");
		}

		alEffecti(Effect, AL_EFFECT_TYPE, AL_EFFECT_CHORUS);
		if (alGetError() == AL_NO_ERROR){
			printf("\n Chorus.....................Yes");
		}
		else{
			printf("\n Chorus.....................No");
		}

		alEffecti(Effect, AL_EFFECT_TYPE, AL_EFFECT_DISTORTION);
		if (alGetError() == AL_NO_ERROR){
			printf("\n Distortion.................Yes");
		}
		else{
			printf("\n Distortion.................No");
		}
```

```
            alEffecti(Effect, AL_EFFECT_TYPE, AL_EFFECT_ECHO);
            if (alGetError() == AL_NO_ERROR){
                  printf("\n Echo.....................Yes");
            }
            else{
                  printf("\n Echo.....................No"); ;
            }
            alEffecti(Effect, AL_EFFECT_TYPE, AL_EFFECT_FLANGER);
            if (alGetError() == AL_NO_ERROR){
                  printf("\n Flanger...................Yes");
            }
            else{
                  printf("\n Flanger...................No");
            }

      alEffecti(Effect,AL_EFFECT_TYPE,AL_EFFECT_FREQUENCY_SHIFTER);
            if (alGetError() == AL_NO_ERROR){
                  printf("\n Frequency Shifter.........Yes");
            }
            else{
                  printf("\n Frequency Shifter.........No");
            }

            alEffecti(Effect, AL_EFFECT_TYPE,AL_EFFECT_VOCAL_MORPHER);
            if (alGetError() == AL_NO_ERROR){
                  printf("\n Vocal Morpher.............Yes");
            }
            else{
                  printf("\n Vocal Morpher.............No");
            }

            alEffecti(Effect, AL_EFFECT_TYPE,AL_EFFECT_PITCH_SHIFTER);
            if (alGetError() == AL_NO_ERROR){
                  printf("\n Pitch Shifter.............Yes");
            }
            else{
                  printf("\n Pitch Shifter.............No");
            }

            alEffecti(Effect,AL_EFFECT_TYPE,AL_EFFECT_RING_MODULATOR);
            if (alGetError() == AL_NO_ERROR){
                  printf("\n Ring Modulator............Yes");
            }
            else{
                  printf("\n Ring Modulator............No");
            }

            alEffecti(Effect, AL_EFFECT_TYPE, AL_EFFECT_AUTOWAH);
            if (alGetError() == AL_NO_ERROR){
                  printf("\n Autowah...................Yes");
            }
            else{
                  printf("\n Autowah...................No");
            }

            alEffecti(Effect, AL_EFFECT_TYPE, AL_EFFECT_COMPRESSOR);
            if (alGetError() == AL_NO_ERROR){
                  printf("\n Compressor................Yes");
            }
```

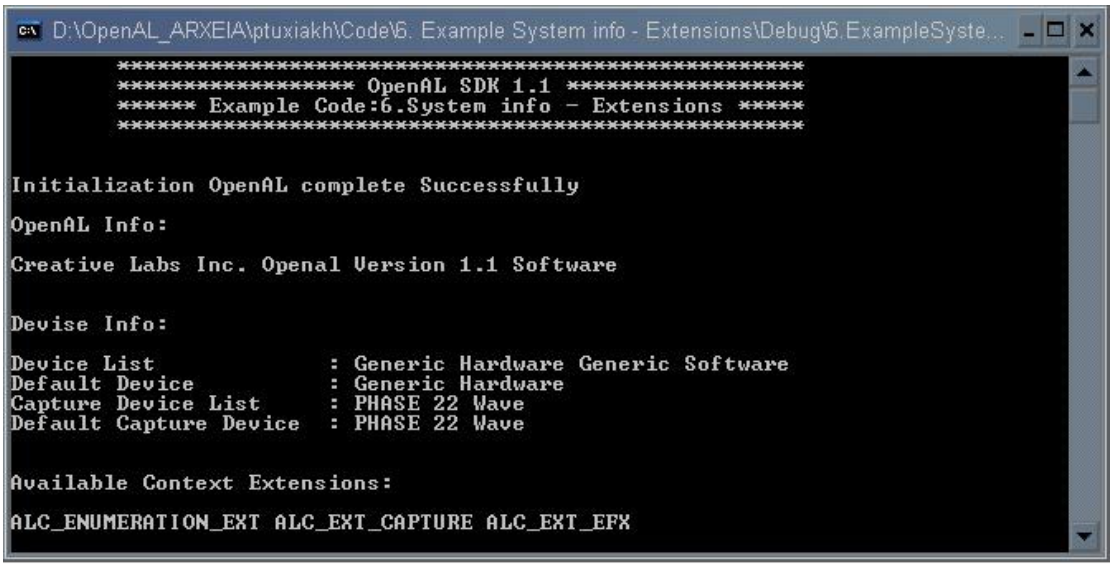
```
            else{
                  printf("\n Compressor.................No");
            }

            alEffecti(Effect, AL_EFFECT_TYPE, AL_EFFECT_EQUALIZER);
            if (alGetError() == AL_NO_ERROR){
                  printf("\n Equalizer..................Yes");
            }
            else{
                  printf("\n Equalizer..................No");
            }
      }
```

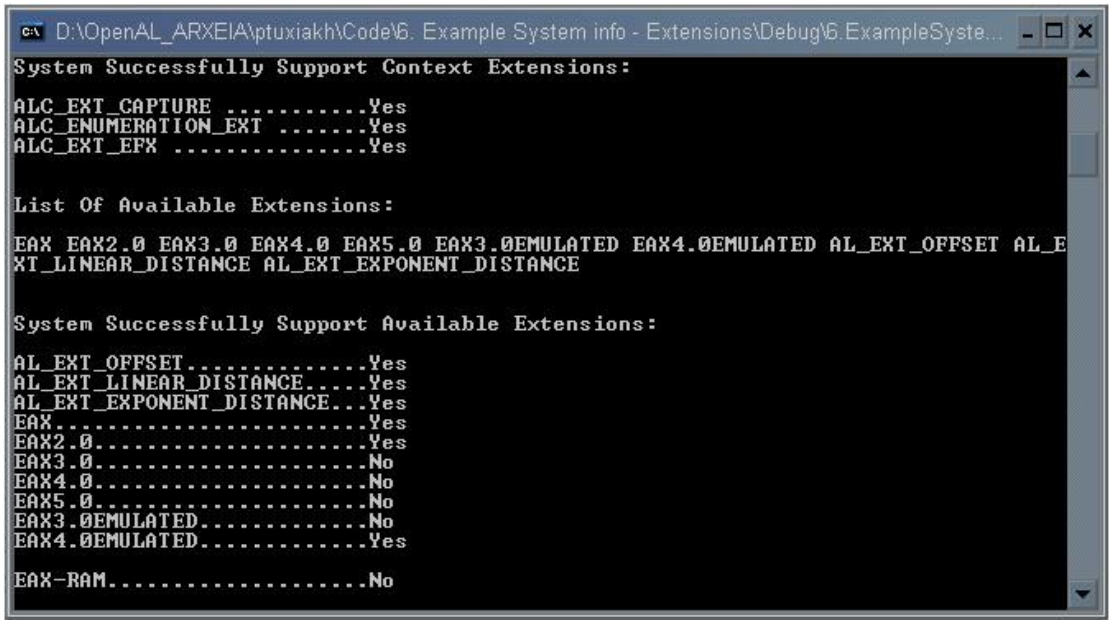
```
      //Test Support Available Filter
      printf("\n\n Device Successfully Support Available Filter: \n");

      alGenFilters(1, &Filter);//Generate 1 Filter Object
      if (alGetError() == AL_NO_ERROR){

            alFilteri(Filter, AL_FILTER_TYPE, AL_FILTER_LOWPASS);
            if (alGetError() == AL_NO_ERROR){
                  printf("\n Low Pass...................Yes");
            }
            else{
                  printf("\n Low Pass...................No");
            }

            alFilteri(Filter, AL_FILTER_TYPE, AL_FILTER_HIGHPASS);
            if (alGetError() == AL_NO_ERROR){
                  printf("\n High Pass..................Yes");
            }
            else{
                  printf("\n High Pass..................No");
            }

            alFilteri(Filter, AL_FILTER_TYPE, AL_FILTER_BANDPASS);
            if (alGetError() == AL_NO_ERROR){
                  printf("\n Band Pass..................Yes");
            }
            else{
                  printf("\n Band Pass..................No");
            }
      }
```

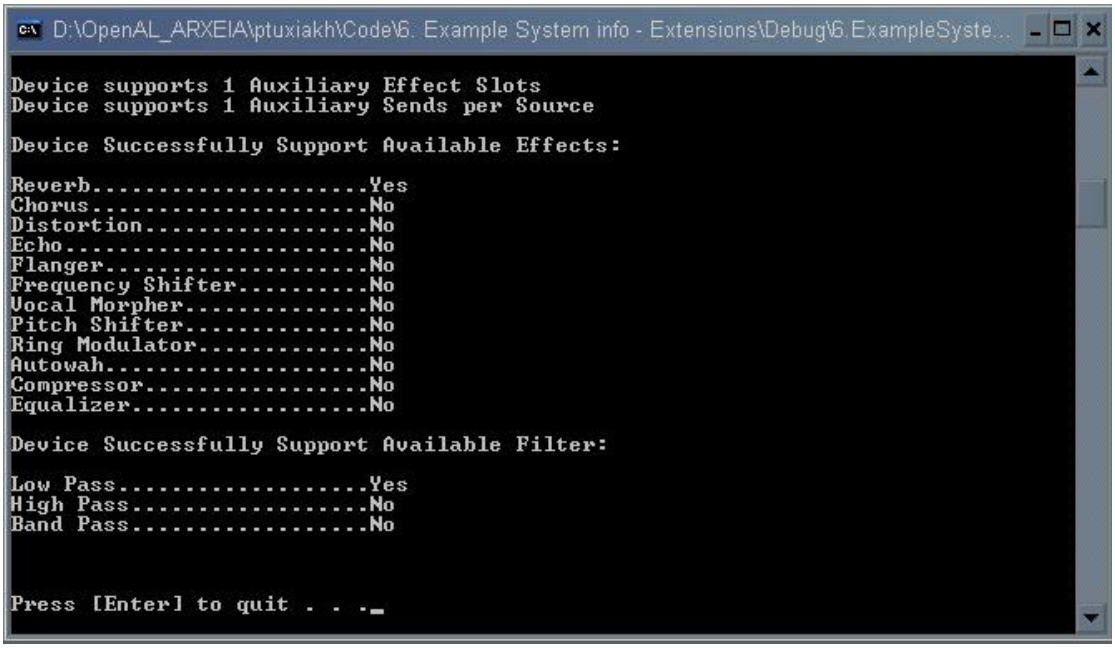
```
      // Delete Filter
      alDeleteFilters(1, &Filter);

      // Delete Effect
      alDeleteEffects(1, &Effect);

      // Delete Auxiliary Effect Slots
      alDeleteAuxiliaryEffectSlots(Loop, EffectSlot);
}
else
{
      printf("EFX not found\n");
}
```

```
////////////////////////////////////////////////////////////////
/******************** OpenAL Exit  ************************/
pContext = alcGetCurrentContext();//open the current context
pDevice = alcGetContextsDevice(pContext);//open a context's device
pointer
alcMakeContextCurrent(NULL);//makes NULL context the current context
alcDestroyContext(pContext);//destroys a context
alcCloseDevice(pDevice);//closes a device

printf ( "\n\n\n\n Press [Enter] to quit . . ." );
fflush ( stdout );
getchar();
return 0;
}
```

Τα αποτελέσματα του συγκεκριμένου παραδείγματος διαφέρουν, ανάλογα με την κάρτα ήχου που διαθέτει ο χρήστης :






D:\OpenAL_ARXEIA\ptuxiakh\Code\6. Example System info - Extensions\Debug\6.ExampleSyste...
Device supports 1 Auxiliary Effect Slots
Device supports 1 Auxiliary Sends per Source
Device Successfully Support Available Effects:
Reverb.....Yes
Chorus.....No
Distortion.....No
Echo.....No
Flanger.....No
Frequency Shifter.....No
Vocal Morpher.....No
Pitch Shifter.....No
Ring Modulator.....No
Autowah.....No
Compressor.....No
Equalizer.....No
Device Successfully Support Available Filter:
Low Pass.....Yes
High Pass.....No
Band Pass.....No
Press [Enter] to quit . . .

7. Example: EFX

Στο συγκεκριμένο παράδειγμα εκτελούνται οι βασικές λειτουργίες της EFX, όπως έχουν προαναφερθεί στο Κεφαλαίο 2.5, για μια τυπική κάρτα ήχου (EAX 2). Αρχικά ένας ήχος εκτελείται χωρίς καμία απολύτως επεξεργασία. Στην συνέχεια ο ίδιος ήχος εκτελείται με ένα Lowpass filter τοποθετημένο στον source. Έπειτα ο ήχος εκτελείται από έναν Auxiliary Effect Slots, που έχει τοποθετημένο πάνω του ένα reverb και χωρίς κανένα filter. Τέλος ο ήχος εκτελείται από ένα Auxiliary Effect Slots, που έχει τοποθετημένο πάνω του ένα reverb και ένα Auxiliary Slots Lowpass filter. Για να είναι εμφανής όλη η λειτουργία, μετά την εκτέλεση του ήχου αποδεσμεύονται όλες οι λειτουργίες που πραγματοποιήθηκαν, ακόμα και όταν την επόμενη φορά που θα εκτελεστεί επαναχρησιμοποιούνται. Έτσι πριν την κάθε εκτέλεση του ήχου τοποθετούνται όλες οι παράμετροι που απαιτούνται, χωρίς να λαμβάνονται υπόψη οι προηγούμενες, μιας και έχουν αποδεσμευτεί.

```
////////////////////////////////////////////////////////////////
/*
 *                      OpenAL SDK 1.1
 *                  Example Code:7.EFX
 *
 *          1.Initializing/Exiting OpenAL and Alut
 *          2.EFX Function setup
 *          3.Load a sound file
 *          4.Generate a Source, Attach Source to Buffer
 *          5.Generate Filter Object, Effect Object
 *            and Auxiliary Effect Slots
 *          6.Play the source(dry)
 *          7.Play the source with Lowpass filter
 *          8.Play the source with auxiliary reverb and no filter
 *          9.Play the source with auxiliary reverb
 *            and auxiliary Lowpass filter
 *          10.Clean up
 *
 */
////////////////////////////////////////////////////////////////
```

/*Include.h*/

```
#ifndef INCLUDE_H_
#define INCLUDE_H_
/*Get some classic includes*/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <Windows.h>
/*This include work ONLY MinGW. NOT compiler work Cygwin...(?) */
#include <conio.h>
/*includes FULL OpenAL*/
#include <al.h>
#include <alc.h>
#include <alut.h>
#include <efx.h>
#include <efx-creative.h>
#include <xram.h>
```

```
/************** GENERAL FUNCTION **************/
void whitePrint(const char *Text);/*Print White Text*/
void RedPrint(const char *Text);/*Print Red  Text */
/*Print Red AL-ALC Error code & Text */
void RedALErrorPrint(const char *Text, ALenum Error);
/*Print Red ALUT Error code & Text */
void RedALUTErrorPrint(const char *Text, ALenum Error);
void ExitEnterPress(void); /*quit programme*/
void StartPrint();/*Start program print*/

/****** INITIALIZATION & EXIT FUNCTION ******/
ALboolean InitOpenAL();/*Initialization OpenAL*/
ALboolean CloseOpenAL();/*Exit-Close OpenAL*/
ALboolean InitALUT();/*Initialization ALUT*/
ALboolean CloseALUT ();/*Exit-Close ALUT*/

#endif /*INCLUDE_H_*/
```

/*Include.c*/

```
/*Include Function File*/

#include "include.h"

/************** GENERAL FUNCTION **************/

void StartPrint(){/*Start program print*/

HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
SetConsoleTextAttribute(h, FOREGROUND_RED | FOREGROUND_GREEN |
                        FOREGROUND_BLUE |FOREGROUND_INTENSITY);

printf ( "*************************************************\n" );
printf ( "****************** OpenAL SDK 1.1 ******************\n" );
printf ( "***************** Example Code:7.EFX ****************\n"
printf ( "*************************************************\n" );
      return ;
}
```

```
/* Colour info:
 * Win32 Console Applications Tutorials- Part 4 colour.
 * http://www.adrianxw.dk/SoftwareSite/Consoles/Consoles4.html*/

void whitePrint(const char *Text)/*Print White Text*/
{
      HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
      SetConsoleTextAttribute(h, FOREGROUND_RED | FOREGROUND_GREEN |
                              FOREGROUND_BLUE |FOREGROUND_INTENSITY);
      printf("%s", Text);
}
void RedPrint(const char *Text)/*Print Red  Text */
{
      HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
      SetConsoleTextAttribute(h, FOREGROUND_RED
                              |FOREGROUND_INTENSITY);
      printf("%s", Text);
}
```

```
/*Print Red AL-ALC Error code & Text */
void RedALErrorPrint(const char *Text, ALenum Error)
{
	HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );

	SetConsoleTextAttribute(h, FOREGROUND_RED
				|FOREGROUND_INTENSITY);
	printf("%s%s\n",Text, alGetString(Error));
}

/*Print Red ALUT Error code & Text */
void RedALUTErrorPrint(const char *Text, ALenum Error)
{
	HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );

	SetConsoleTextAttribute(h, FOREGROUND_RED
				|FOREGROUND_INTENSITY);
	printf("%s%s\n",Text, alutGetErrorString(Error));
}
```

```
void ExitEnterPress(void) /*quit programme*/
{
	whitePrint ( "\nPress [Enter] to quit . . ." );
	fflush ( stdout );
	getchar();
	exit(0); /*Returns 0 to the operating system*/
}
```

```
/****** INITIALIZATION & EXIT FUNCTION ******/
ALboolean InitOpenAL()/*Initialization OpenAL manually*/
{
ALCcontext	*pContext = NULL;
ALCdevice	*pDevice = NULL;
ALboolean	bReturn = AL_FALSE;
ALenum		Error; /* save error status*/
ALCboolean	currentcon ;/*test for alcMakeContextCurrent*/

pDevice = alcOpenDevice(NULL); /*Open default device*/
if (pDevice){ /*pDevice != NULL*/
	alcGetError(pDevice);/*Clear error code*/
	/*creates a context using a default device*/
	pContext = alcCreateContext(pDevice, NULL);
	if (pContext){/*pContext != NULL*/
	 /*set active context-Makes a pContext the current context*/
		currentcon = alcMakeContextCurrent(pContext);
		if (currentcon == ALC_TRUE){
			bReturn = AL_TRUE;
		}
		else{/*alcMakeContextCurrent(pContext)!= ALC_TRUE */
			Error = alcGetError(pDevice);
			RedALErrorPrint( "Unable to make OpenAL context
					current: !!!Error!!!", Error );
		}
	}
	else {/*pContext = NULL*/
		Error = alcGetError(pDevice);/*context error state*/
			RedALErrorPrint ( "Unable to create an OpenAL
				context: !!!Error!!! ",Error);
		}
```

```
	}
	else{/*pDevice = NULL*/

		HANDLE h = GetStdHandle ( STD_OUTPUT_HANDLE );
		SetConsoleTextAttribute(h, FOREGROUND_RED
					|FOREGROUND_INTENSITY);
		printf("Unable to Open default device :!!!Error!!!");

	}
	return bReturn;
}
```

```
ALboolean CloseOpenAL()/*Exit-Close OpenAL*/
{
ALCcontext *pContext;
ALCdevice *pDevice;
ALCboolean currentcon ;/*test for alcMakeContextCurrent*/

pContext = alcGetCurrentContext(); /*open the current context */
if (alcGetCurrentContext() == NULL){
	whitePrint ( "\n WARNING: There is no current context, NULL is
returned........" );
}
/*open a context's device pointer*/
pDevice = alcGetContextsDevice(pContext);
/*makes NULL context the current context*/
currentcon = alcMakeContextCurrent(NULL);
if (currentcon != ALC_TRUE){
	whitePrint ( "WARNING: failure to makes NULL context the
		current context......" );
}
alcDestroyContext(pContext);/*destroys a context*/
alcCloseDevice(pDevice);/*closes a device*/
return AL_TRUE;
}
```

```
ALboolean  InitALUT()/*Initialization ALUT*/
{
	ALboolean	bReturn = AL_FALSE;
	ALenum		Error;		/*Save ALUT error status*/

	alutGetError();/*Clear error code*/

	if (!alutInitWithoutContext( NULL , NULL )) {
		Error = alutGetError();
		RedALUTErrorPrint("!!!Error!!!", Error);
	}
	else
	{
	bReturn = AL_TRUE;
	}
	return bReturn;
}
```

```
ALboolean CloseALUT()/*Exit-Close ALUT */
{
	ALboolean	bReturn = AL_FALSE;
	ALenum		Error; /*Save ALUT error status*/

	alutGetError();/*Clear error code*/
```

```
	if (!alutExit()) {
		Error = alutGetError();
		RedALUTErrorPrint("Close ALUT : !!!Error!!!", Error);
	}
	bReturn = AL_TRUE;
	return bReturn;
}
```

/* Main.c*/

```
/*Include Function File*/
#include "include.h"

// Effect objects
LPALGENEFFECTS alGenEffects = NULL;
LPALDELETEEFFECTS alDeleteEffects = NULL;
LPALISEFFECT alIsEffect = NULL;
LPALEFFECTI alEffecti = NULL;
LPALEFFECTIV alEffectiv = NULL;
LPALEFFECTF alEffectf = NULL;
LPALEFFECTFV alEffectfv = NULL;
LPALGETEFFECTI alGetEffecti = NULL;
LPALGETEFFECTIV alGetEffectiv = NULL;
LPALGETEFFECTF alGetEffectf = NULL;
LPALGETEFFECTFV alGetEffectfv = NULL;


//Filter objects
LPALGENFILTERS alGenFilters = NULL;
LPALDELETEFILTERS alDeleteFilters = NULL;
LPALISFILTER alIsFilter = NULL;
LPALFILTERI alFilteri = NULL;
LPALFILTERIV alFilteriv = NULL;
LPALFILTERF alFilterf = NULL;
LPALFILTERFV alFilterfv = NULL;
LPALGETFILTERI alGetFilteri = NULL;
LPALGETFILTERIV alGetFilteriv = NULL;
LPALGETFILTERF alGetFilterf = NULL;
LPALGETFILTERFV alGetFilterfv = NULL;


// Auxiliary slot object
LPALGENAUXILIARYEFFECTSLOTS alGenAuxiliaryEffectSlots = NULL;
LPALDELETEAUXILIARYEFFECTSLOTS alDeleteAuxiliaryEffectSlots = NULL;
LPALISAUXILIARYEFFECTSLOT alIsAuxiliaryEffectSlot = NULL;
LPALAUXILIARYEFFECTSLOTI alAuxiliaryEffectSloti = NULL;
LPALAUXILIARYEFFECTSLOTIV alAuxiliaryEffectSlotiv = NULL;
LPALAUXILIARYEFFECTSLOTF alAuxiliaryEffectSlotf = NULL;
LPALAUXILIARYEFFECTSLOTFV alAuxiliaryEffectSlotfv = NULL;
LPALGETAUXILIARYEFFECTSLOTI alGetAuxiliaryEffectSloti = NULL;
LPALGETAUXILIARYEFFECTSLOTIV alGetAuxiliaryEffectSlotiv = NULL;
LPALGETAUXILIARYEFFECTSLOTF alGetAuxiliaryEffectSlotf = NULL;
LPALGETAUXILIARYEFFECTSLOTFV alGetAuxiliaryEffectSlotfv = NULL;
```

```
//EFX Function setup
ALboolean EFXFunctionSetup()
{
ALCdevice *pDevice = NULL;
ALCcontext *pContext = NULL;
ALboolean   Setup = AL_FALSE;

pContext = alcGetCurrentContext();
pDevice = alcGetContextsDevice(pContext);

if (alcIsExtensionPresent(pDevice, (ALCchar*)ALC_EXT_EFX_NAME))
{
// Get function pointers
alGenEffects = (LPALGENEFFECTS)alGetProcAddress("alGenEffects");
alDeleteEffects = (LPALDELETEEFFECTS )alGetProcAddress("alDeleteEffects");
alIsEffect = (LPALISEFFECT )alGetProcAddress("alIsEffect");
alEffecti = (LPALEFFECTI)alGetProcAddress("alEffecti");
alEffectiv = (LPALEFFECTIV)alGetProcAddress("alEffectiv");
alEffectf = (LPALEFFECTF)alGetProcAddress("alEffectf");
alEffectfv = (LPALEFFECTFV)alGetProcAddress("alEffectfv");
alGetEffecti = (LPALGETEFFECTI)alGetProcAddress("alGetEffecti");
alGetEffectiv = (LPALGETEFFECTIV)alGetProcAddress("alGetEffectiv");
alGetEffectf = (LPALGETEFFECTF)alGetProcAddress("alGetEffectf");
alGetEffectfv = (LPALGETEFFECTFV)alGetProcAddress("alGetEffectfv");

alGenFilters = (LPALGENFILTERS)alGetProcAddress("alGenFilters");
alDeleteFilters = (LPALDELETEFILTERS)alGetProcAddress("alDeleteFilters");
alIsFilter = (LPALISFILTER)alGetProcAddress("alIsFilter");
alFilteri = (LPALFILTERI)alGetProcAddress("alFilteri");
alFilteriv = (LPALFILTERIV)alGetProcAddress("alFilteriv");
alFilterf = (LPALFILTERF)alGetProcAddress("alFilterf");
alFilterfv = (LPALFILTERFV)alGetProcAddress("alFilterfv");
alGetFilteri = (LPALGETFILTERI )alGetProcAddress("alGetFilteri");
alGetFilteriv= (LPALGETFILTERIV )alGetProcAddress("alGetFilteriv");
alGetFilterf = (LPALGETFILTERF )alGetProcAddress("alGetFilterf");
alGetFilterfv= (LPALGETFILTERFV )alGetProcAddress("alGetFilterfv");

alGenAuxiliaryEffectSlots = (LPALGENAUXILIARYEFFECTSLOTS)alGetProcAddress
                            ("alGenAuxiliaryEffectSlots");
alDeleteAuxiliaryEffectSlots = (LPALDELETEAUXILIARYEFFECTSLOTS)
                          alGetProcAddress("alDeleteAuxiliaryEffectSlots");
alIsAuxiliaryEffectSlot = (LPALISAUXILIARYEFFECTSLOT)alGetProcAddress
                          ("alIsAuxiliaryEffectSlot");
alAuxiliaryEffectSloti = (LPALAUXILIARYEFFECTSLOTI)alGetProcAddress
                         ("alAuxiliaryEffectSloti");
alAuxiliaryEffectSlotiv = (LPALAUXILIARYEFFECTSLOTIV)alGetProcAddress
                          ("alAuxiliaryEffectSlotiv");
alAuxiliaryEffectSlotf = (LPALAUXILIARYEFFECTSLOTF)alGetProcAddress
                         ("alAuxiliaryEffectSlotf");
alAuxiliaryEffectSlotfv = (LPALAUXILIARYEFFECTSLOTFV)alGetProcAddress
                          ("alAuxiliaryEffectSlotfv");
alGetAuxiliaryEffectSloti = (LPALGETAUXILIARYEFFECTSLOTI)alGetProcAddress
                            ("alGetAuxiliaryEffectSloti");
alGetAuxiliaryEffectSlotiv = (LPALGETAUXILIARYEFFECTSLOTIV)alGetProcAddress
                             ("alGetAuxiliaryEffectSlotiv");
alGetAuxiliaryEffectSlotf = (LPALGETAUXILIARYEFFECTSLOTF)alGetProcAddress
                            ("alGetAuxiliaryEffectSlotf");
alGetAuxiliaryEffectSlotfv = (LPALGETAUXILIARYEFFECTSLOTFV)alGetProcAddress
                             ("alGetAuxiliaryEffectSlotfv");
```

```
      if (alGenEffects && alDeleteEffects && alIsEffect && alEffecti
          && alEffectiv &&alEffectf && alEffectfv && alGetEffecti &&
          alGetEffectiv && alGetEffectf && alGetEffectfv &&
          alGenFilters && alDeleteFilters && alIsFilter && alFilteri
         && alFilteriv && alFilterf && alFilterfv && alGetFilteri &&
          alGetFilteriv && alGetFilterf && alGetFilterfv &&
          alGenAuxiliaryEffectSlots &&
          alDeleteAuxiliaryEffectSlots && alIsAuxiliaryEffectSlot &&
          alAuxiliaryEffectSloti && alAuxiliaryEffectSlotiv &&
          alAuxiliaryEffectSlotf && alAuxiliaryEffectSlotfv &&
          alGetAuxiliaryEffectSloti && alGetAuxiliaryEffectSlotiv &&
          alGetAuxiliaryEffectSlotf && alGetAuxiliaryEffectSlotfv)
            Setup = AL_TRUE;
      }

return Setup;
}
```

```
int main()
{
//////////////////////////////////////////////////////////
/******************* Definition Variable ******************/

ALCenum     Error; //save error status
ALuint      Buffer;//Buffer Object
ALuint      Source;//source Object
ALuint      Effect;//Effect Object
ALuint      Filter;//Filter Object
ALuint      EffectSlot;//Auxiliary Effect Slots Object
ALint       state;
ALboolean   AReturn     = AL_FALSE; // TRUE OR FALSE Function  Return
ALboolean   BReturn     = AL_FALSE; // TRUE OR FALSE Function  Return

const char  *FileName   = "sounds\\coca_kola.wav";//Sound Path
:..../sounds/
```

```
//////////////////////////////////////////////////////////
/*************** Initializing OpenAL  ********************/

StartPrint(); //Start program print
AReturn   = InitOpenAL();
BReturn   = InitALUT();
if ( AReturn == AL_TRUE && BReturn == AL_TRUE ){
      whitePrint( "\n Initialization OpenAL Complete:\n\n" );
}
else {
      RedPrint("\n !ERROR! Initialization OpenAL NOT Complete\n\n");
      CloseALUT();
      CloseOpenAL();
      ExitEnterPress();
}

// Check for EFX Extension
if(!EFXFunctionSetup()){
      RedPrint("\n\n EFX Extension not found\n");
      CloseALUT();
      CloseOpenAL();
      ExitEnterPress();
}
```

```
////////////////////////////////////////////////////////////////
/******************* MAIN PROGRAM ************************/
alutGetError();   //Clear ALUT  Error State
alGetError();// Clear AL Error State

Buffer=alutCreateBufferFromFile ( FileName );//Load a .WAV sound
if (Buffer == AL_NONE ){
      Error = alutGetError();
      RedALUTErrorPrint("\n !!!Error!!! loading file: ",Error);
      printf("Failed to load %s\n", FileName);
      alDeleteBuffers(1,&Buffer);//delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}

alGenSources( 1, &Source );//Generate source object
Error = alGetError ();
if ( Error != AL_NO_ERROR ){
      RedALErrorPrint("\n!!!Error!!! Failure to Generate a Source:",
Error);
      alDeleteSources(1, &Source);//delete source
      alDeleteBuffers(1,&Buffer); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}

alSourcei (Source, AL_BUFFER, Buffer);//Attach Source to Buffer
Error = alGetError ();
if ( Error != AL_NO_ERROR ){
      RedALErrorPrint("\n!!!Error!!! Failure to Attach Source to
                       Buffer:" ,Error);
      alDeleteSources(1,&Source);//delete source
      alDeleteBuffers(1,&Buffer); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}

alGenFilters(1, &Filter);//Generate 1 Filter Object
Error = alGetError ();
if ( Error != AL_NO_ERROR ){
      RedALErrorPrint("\n!!!Error!!! Failure to Generate a Filter:
",Error);
      alDeleteFilters(1, &Filter);// Delete Filter
      alDeleteSources(1, &Source);//delete source
      alDeleteBuffers(1,&Buffer); //delete Buffer
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitEnterPress();//quit programme
}

alGenEffects(1, &Effect);//Generate 1 Effect Object
Error = alGetError ();
if ( Error != AL_NO_ERROR ){
      RedALErrorPrint("\n !!!Error!!! Failure to Generate a
Effect:",Error);
      alDeleteEffects(1, &Effect);// Delete Effect
```

```
	alDeleteFilters(1, &Filter);// Delete Filter
	alDeleteSources(1, &Source);//delete source
	alDeleteBuffers(1,&Buffer); //delete Buffer
	CloseALUT();//Exit-Close ALUT
	CloseOpenAL();//Exit-Close OpenAL
	ExitEnterPress();//quit programme
}

//Generate 1 Auxiliary Effect Slots
alGenAuxiliaryEffectSlots( 1, &EffectSlot );
Error = alGetError ();
if ( Error != AL_NO_ERROR ){
	RedALErrorPrint("\n !!!Error!!! Failure to Generate a Auxiliary
Effect
                        Slots:", Error);
	//Delete Auxiliary Effect Slots
	alDeleteAuxiliaryEffectSlots(1, &EffectSlot);
	alDeleteEffects(1, &Effect);// Delete Effect
	alDeleteFilters(1, &Filter);// Delete Filter
	alDeleteSources(1, &Source);//delete source
	alDeleteBuffers(1,&Buffer); //delete Buffer
	CloseALUT();//Exit-Close ALUT
	CloseOpenAL();//Exit-Close OpenAL
	ExitEnterPress();//quit programme
}
```

```
//1.Play the source
whitePrint( "\n1.Play the source (dry)\n" );
alSourcePlay( Source );

while( 1  ){
	alGetSourcei( Source, AL_SOURCE_STATE, &state); // Get state
	if( state != AL_PLAYING )
		break;
	}
```

```
//2.Play the source with Lowpass filter
whitePrint( "2.Play the source with Lowpass filter\n" );

alFilteri(Filter, AL_FILTER_TYPE, AL_FILTER_LOWPASS);
if ( Error != AL_NO_ERROR ){
		RedALErrorPrint("\n !!!Error!!! Failure to Attach Lowpass
filter
                                to filter: ",Error);
		//Delete Auxiliary Effect Slots
		alDeleteAuxiliaryEffectSlots(1, &EffectSlot);
		alDeleteEffects(1, &Effect);// Delete Effect
		alDeleteFilters(1, &Filter);// Delete Filter
		alDeleteSources(1,&Source);//delete source
		alDeleteBuffers(1,&Buffer); //delete Buffer
		CloseALUT();//Exit-Close ALUT
		CloseOpenAL();//Exit-Close OpenAL
		ExitEnterPress();//quit programme
	}
alFilterf(Filter, AL_LOWPASS_GAIN, 1.0f);
alFilterf(Filter, AL_LOWPASS_GAINHF, 0.2f);

alSourcei(Source, AL_DIRECT_FILTER, Filter);
```

```
if ( Error != AL_NO_ERROR ){
            RedALErrorPrint("\n !!!Error!!! Failure to Attach Lowpass
filter
                              to Source: ",Error);
            //Delete Auxiliary Effect Slots
            alDeleteAuxiliaryEffectSlots(1, &EffectSlot);
            alDeleteEffects(1, &Effect);// Delete Effect
            alDeleteFilters(1, &Filter);// Delete Filter
            alDeleteSources(1,&Source);//delete source
            alDeleteBuffers(1,&Buffer); //delete Buffer
            CloseALUT();//Exit-Close ALUT
            CloseOpenAL();//Exit-Close OpenAL
            ExitEnterPress();//quit programme
}
            alSourcePlay( Source );

            while( 1  ){
                  //Get state
                  alGetSourcei( Source,AL_SOURCE_STATE, &state);
                  if( state != AL_PLAYING )
                        break;
                  }

             //Remove a Filter from Source
             alSourcei(Source, AL_DIRECT_FILTER, AL_FILTER_NULL);
```

```
//3.Play the source with auxiliary reverb and no filter
whitePrint("3.Play the source with auxiliary reverb and no
            filter\n");

alEffecti( Effect, AL_EFFECT_TYPE, AL_EFFECT_REVERB );
if ( Error != AL_NO_ERROR ){
       RedALErrorPrint("\n !!!Error!!! Failure to Attach Reverb to
                        effect: ",Error);
            //Delete Auxiliary Effect Slots
            alDeleteAuxiliaryEffectSlots(1, &EffectSlot);
            alDeleteEffects(1, &Effect);// Delete Effect
            alDeleteFilters(1, &Filter);// Delete Filter
            alDeleteSources(1,&Source);//delete source
            alDeleteBuffers(1,&Buffer); //delete Buffer
            CloseALUT();//Exit-Close ALUT
            CloseOpenAL();//Exit-Close OpenAL
            ExitEnterPress();//quit programme
}

alAuxiliaryEffectSloti( EffectSlot, AL_EFFECTSLOT_EFFECT, Effect );
if ( Error != AL_NO_ERROR ){
       RedALErrorPrint("\n !!!Error!!! Failure to Attach Reverb
                        effect to Auxiliary Effect Slots: ",Error);
                  //Delete Auxiliary Effect Slots
                  alDeleteAuxiliaryEffectSlots(1, &EffectSlot);
                  alDeleteEffects(1, &Effect);// Delete Effect
                  alDeleteFilters(1, &Filter);// Delete Filter
                  alDeleteSources(1,&Source);//delete source
                  alDeleteBuffers(1,&Buffer); //delete Buffer
                  CloseALUT();//Exit-Close ALUT
                  CloseOpenAL();//Exit-Close OpenAL
                  ExitEnterPress();//quit programme
}
```

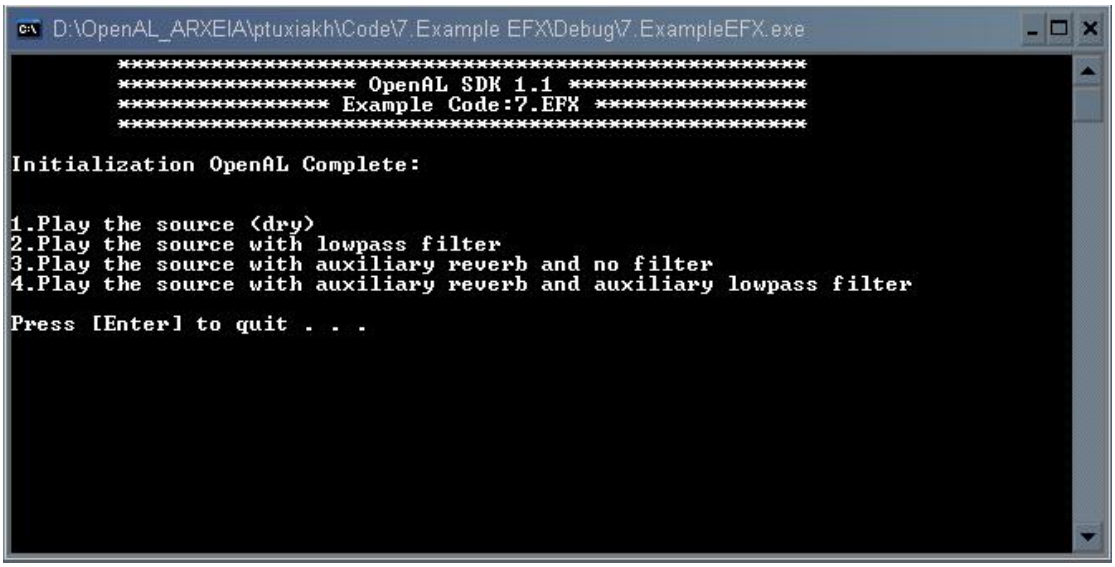
```
alSource3i(Source, AL_AUXILIARY_SEND_FILTER, EffectSlot, 0,
AL_FILTER_NULL);
 if ( Error != AL_NO_ERROR ){
       RedALErrorPrint("\n!!!Error!!! Failure to Attach Source to
                        Auxiliary Reverb Effect Slots: ",Error);
            //Delete Auxiliary Effect Slots
            alDeleteAuxiliaryEffectSlots(1, &EffectSlot);
            alDeleteEffects(1, &Effect);// Delete Effect
            alDeleteFilters(1, &Filter);// Delete Filter
            alDeleteSources(1,&Source);//delete source
            alDeleteBuffers(1,&Buffer); //delete Buffer
            CloseALUT();//Exit-Close ALUT
            CloseOpenAL();//Exit-Close OpenAL
            ExitEnterPress();//quit programme
}

alSourcePlay( Source );

while( 1  ){
      alGetSourcei( Source, AL_SOURCE_STATE, &state); // Get state
      if( state != AL_PLAYING )
            break;
      }

//Remove a Source from Auxiliary Effect Slots
alSource3i(Source, AL_AUXILIARY_SEND_FILTER,
AL_EFFECTSLOT_NULL,0,AL_FILTER_NULL);
```

```
//4.Play the source with auxiliary reverb and auxiliary  Lowpass
filter
whitePrint( "4.Play the source with auxiliary reverb and auxiliary
            Lowpass filter\n" );

alFilteri(Filter, AL_FILTER_TYPE, AL_FILTER_LOWPASS);
if ( Error != AL_NO_ERROR ){
      RedALErrorPrint("\n !!!Error!!! Failure to Attach Lowpass
                       filter to filter: ",Error);
            //Delete Auxiliary Effect Slots
            alDeleteAuxiliaryEffectSlots(1, &EffectSlot);
            alDeleteEffects(1, &Effect);// Delete Effect
            alDeleteFilters(1, &Filter);// Delete Filter
            alDeleteSources(1,&Source);//delete source
            alDeleteBuffers(1,&Buffer); //delete Buffer
            CloseALUT();//Exit-Close ALUT
            CloseOpenAL();//Exit-Close OpenAL
            ExitEnterPress();//quit programme
}

alFilterf(Filter, AL_LOWPASS_GAIN, 1.0f);

alFilterf(Filter, AL_LOWPASS_GAINHF, 0.2f);

alEffecti( Effect, AL_EFFECT_TYPE, AL_EFFECT_REVERB );
 if ( Error != AL_NO_ERROR ){
      RedALErrorPrint("\n !!!Error!!! Failure to Attach Reverb
                       to effect: ",Error);
```

```
                    //Delete Auxiliary Effect Slots
                    alDeleteAuxiliaryEffectSlots(1, &EffectSlot);
                    alDeleteEffects(1, &Effect);// Delete Effect
                    alDeleteFilters(1, &Filter);// Delete Filter
                    alDeleteSources(1,&Source);//delete source
                    alDeleteBuffers(1,&Buffer); //delete Buffer
                    CloseALUT();//Exit-Close ALUT
                    CloseOpenAL();//Exit-Close OpenAL
                    ExitEnterPress();//quit programme
}


alAuxiliaryEffectSloti( EffectSlot, AL_EFFECTSLOT_EFFECT, Effect );
if ( Error != AL_NO_ERROR ){
      RedALErrorPrint("\n !!!Error!!! Failure to Attach Reverb
                        effect to Auxiliary Effect Slots: ",Error);
            //Delete Auxiliary Effect Slots
            alDeleteAuxiliaryEffectSlots(1, &EffectSlot);
            alDeleteEffects(1, &Effect);// Delete Effect
            alDeleteFilters(1, &Filter);// Delete Filter
            alDeleteSources(1,&Source);//delete source
            alDeleteBuffers(1,&Buffer); //delete Buffer
            CloseALUT();//Exit-Close ALUT
            CloseOpenAL();//Exit-Close OpenAL
            ExitEnterPress();//quit programme
}
alSource3i(Source, AL_AUXILIARY_SEND_FILTER, EffectSlot, 0, Filter);

if ( Error != AL_NO_ERROR ){
      RedALErrorPrint("\n !!!Error!!! Failure to Attach Source to
                        Auxiliary Effect Slots  and auxiliary Lowpass
                        filter ",Error);
            //Delete Auxiliary Effect Slots
            alDeleteAuxiliaryEffectSlots(1, &EffectSlot);
            alDeleteEffects(1, &Effect);// Delete Effect
            alDeleteFilters(1, &Filter);// Delete Filter
            alDeleteSources(1,&Source);//delete source
            alDeleteBuffers(1,&Buffer); //delete Buffer
            CloseALUT();//Exit-Close ALUT
            CloseOpenAL();//Exit-Close OpenAL
            ExitEnterPress();//quit programme
}
alSourcePlay( Source );

while( 1  ){
      alGetSourcei( Source, AL_SOURCE_STATE, &state);//Get state
      if( state != AL_PLAYING )
                  break;
      }



//Remove a Source from Auxiliary Effect Slots  and auxiliary
//Lowpass filter

alSource3i(Source, AL_AUXILIARY_SEND_FILTER, AL_EFFECTSLOT_NULL, 0,
AL_FILTER_NULL);
```

```
////////////////////////////////////////////////////////////
/******************** OpenAL Exit  ************************/

alDeleteAuxiliaryEffectSlots(1, &EffectSlot);//Delete Auxiliary
Effect Slots
alDeleteEffects(1, &Effect);// Delete Effect
alDeleteFilters(1, &Filter);// Delete Filter
alDeleteSources(1,&Source);//delete source
alDeleteBuffers(1,&Buffer); //delete Buffer
CloseALUT();//Exit-Close ALUT
CloseOpenAL();//Exit-Close OpenAL
ExitEnterPress();//quit programme

return 0;
}
```

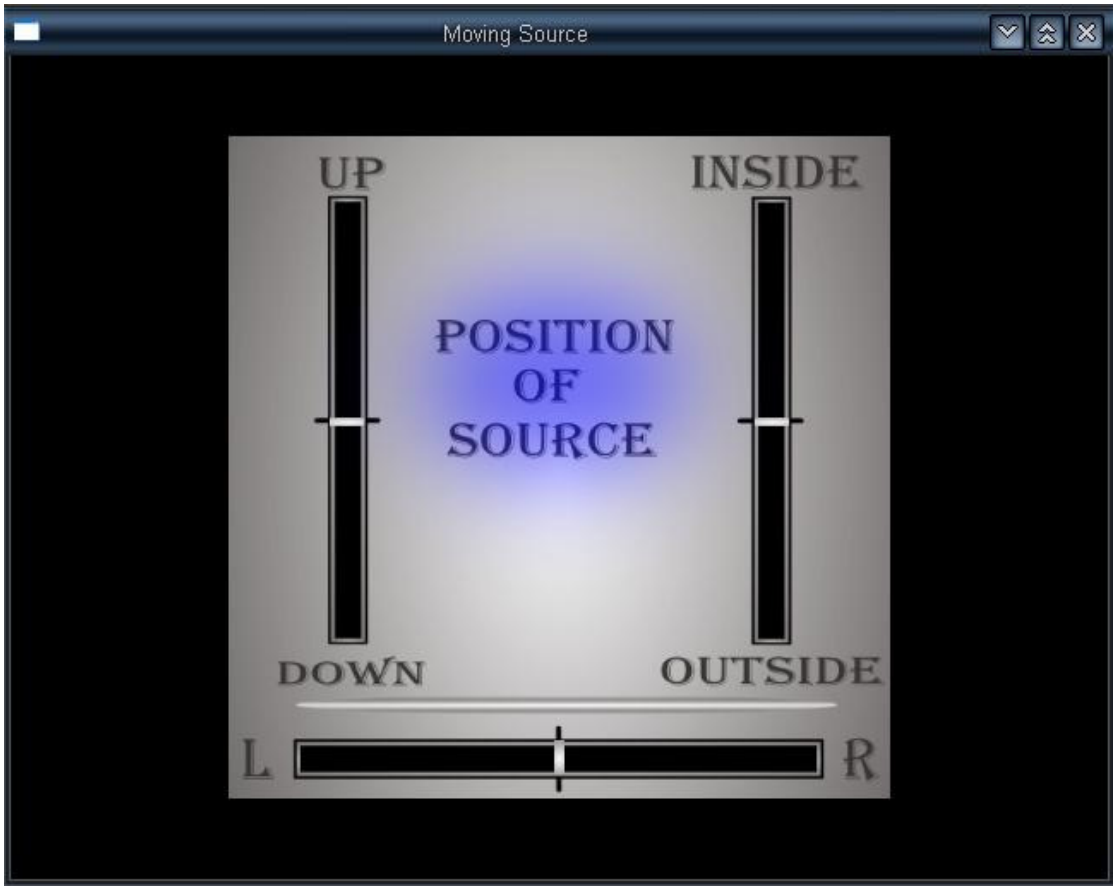
Κατά την εκτέλεση του παραδείγματος εκτυπώνονται στην οθόνη τα ακόλουθα μηνύματα:



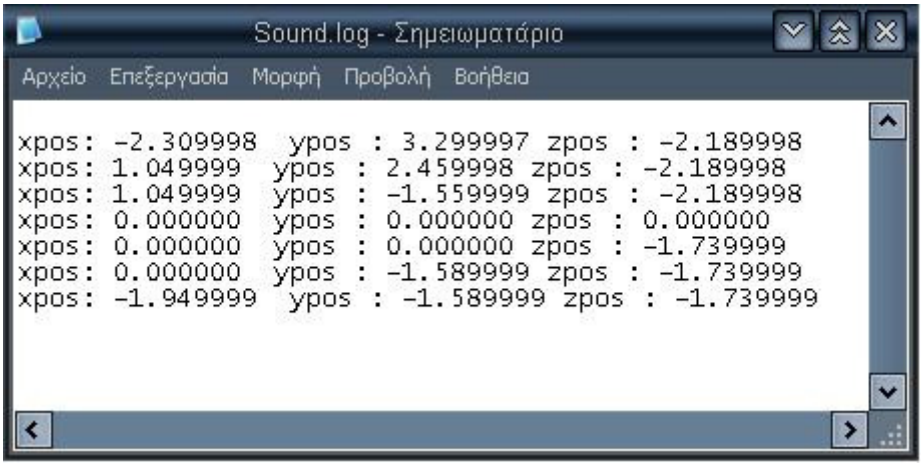
8. Example: GUI - Moving Source

Το παράδειγμα οκτώ αποτελεί ένα γραφικό interface, που χρησιμοποιείται για την μετακίνηση και τοποθέτηση ενός Source στον τρισδιάστατο χώρο. Για την δημιουργία του χρησιμοποιήθηκε η OpenAL, το win32 API και η OpenGL. Είναι γραμμένο σε γλώσσα C++ με τον Microsoft Visual C++ 2008 Express Edition. Ο λόγος για την διαφοροποίηση αυτή έχει άμεση σχέση με το win32 API, μιας και δεν υποστηρίζεται πλήρως από το GCC. Για την δημιουργία του παραδείγματος χρησιμοποιήθηκαν πληροφορίες από τον δικτυακό τόπο του the Forger's Win32 API Tutorial (http://www.winprog.org/tutorial/), ενώ πληροφορίες, καθώς και τμήματα κώδικα, όσον αφορά την δομή του window και της OpenGL, προήλθαν από τον δικτυακό τόπο Nehe Productions (http://nehe.gamedev.net/) Lessons 01 -10.

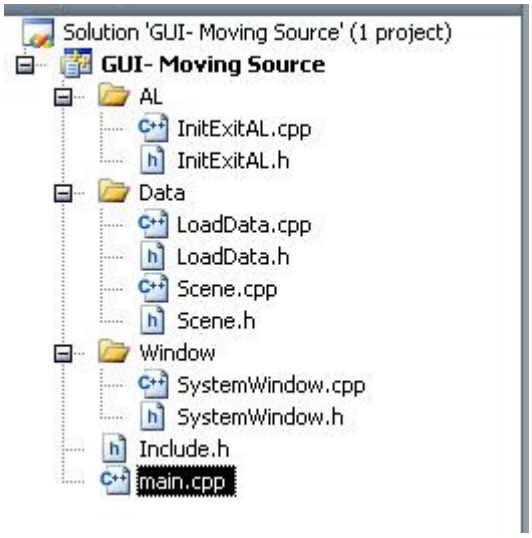
Το παράδειγμα δημιουργεί ένα window σε διαστάσεις 640x480, με ανάλυση 32bit χρωμάτων. Στην συνέχεια εμφανίζεται ένα interface, που μπορει να χρησιμοποηθεί από τον χρήστη με βελάκια και τα Page up-down του πληκτρολογίου και να μετακινήσει τον Source ή να τον τοποθετήσει σε μια συγκεκριμένη θέση.



Ταυτόχρονα με το πλήκτρο C μπορεί να αντιγράψει την θέση που έχει τοποθετήσει τον Source σε ένα αρχείο που ονομάζεται Sound.log.



Στην παρακάτω εικόνα παρουσιάζεται η δομή του κώδικα



Όπως φαίνεται, το πρόγραμμα χωρίζεται σε 3 υποφακέλους. Στον υποφάκελο AL υπάρχουν οι συναρτήσεις για το initialize και exit των AL και ALUT. Στον υποφάκελο Data το αρχείο loadData περιέχει όλες τις συναρτήσεις που απαιτούνται για να γίνουν load τα textures του interface και ο ήχος, ενώ το αρχείο Scene είναι ο πυρήνας όλου του προγράμματος, δηλαδή είναι το μέρος που δημιουργείται το interface, ενώ επίσης γίνονται και όλες οι ηχητικές επεξεργασίες που παρουσιάζονται στο πρόγραμμα. Ουσιαστικά περιέχει όλες τις λειτουργίες που πραγματοποιούνται, αφού δημιουργηθεί το γραφικό περιβάλλον. Στον υποφάκελο window υπάρχουν όλες οι συναρτήσεις που χρειάζονται για την δημιουργία ενός window GL, καθώς και μια συνάρτηση ρύθμισης για το μέγεθος του windows- ReSize windows.

Το πρόγραμμα ουσιαστικά ξεκινάει εκτελώντας την παρακάτω γραμμή εντολών

```
if (!WindowsRAN("Moving Source", 640,480,32)){
            return 0;//Quit If Window Was Not Created
      }
```

Η συνάρτηση αυτή είναι μέσα στο αρχείο systemWindow και εκτελεί την εξής διαδικασία :

- Εκτελεί την συνάρτηση ALSound() από το αρχείο lintExitAL.cpp που εγκαθιστά την OpenAL και ALUT. Στην περίπτωση που η ALSound() επιστρέψει FALSE, τότε το πρόγραμμα σταματάει να εκτελείται, γιατί ή δεν εγκαταστάθηκε σωστά η OpenAL ή η ALUT και επιστρέφει ένα μήνυμα σφάλματος, που ενημερώνει τον χρήστη να ελέγξει την κάρτα ήχου του για τυχόν βλάβες.

- Εκτελεί την συνάρτηση LoadWavSound () από το αρχείο LoadData.cpp, που φορτώνει ένα αρχείο ήχου σε ένα buffer. Στην περίπτωση που η LoadWavSound() επιστρέψει FALSE, τότε το πρόγραμμα σταματάει να εκτελείται και επιστρέφει ένα αντίστοιχο μήνυμα σφάλματος. Σε αυτήν την περίπτωση το πιθανότερο σφάλμα είναι ότι δεν βρέθηκαν τα αρχεία ήχου στο Path : Data\ Sounds\.

- Εκτελεί την συνάρτηση CreateGLWindow(title,width,height,bits) από το αρχείο systemWidow.cpp που δημιουργεί ένα OpenGL window. Στην περίπτωση που η CreateGLWindow(title,width,height,bits) επιστρέψει FALSE, τότε το πρόγραμμα σταματάει να εκτελείται και επιστρέφει ένα αντίστοιχο μήνυμα σφάλματος. Σε αυτήν την περίπτωση το πιθανότερο είναι να χρειάζονται αναβάθμιση οι drivers της κάρτας γραφικών.

- Εκτελεί την συνάρτηση LoadGLTextures() από το αρχείο LoadData.cpp, που φορτώνει δύο αρχεία εικόνας. Στην περίπτωση που η LoadGLTextures() επιστρέψει FALSE, τότε το πρόγραμμα σταματάει να εκτελείται και επιστρέφει ένα αντίστοιχο μήνυμα σφάλματος. Σε αυτήν την περίπτωση το πιθανότερο σφάλμα είναι ότι δεν υπάρχουν τα αρχεία εικόνας στο Path : Data\Textures\.

- Εκτελεί την συνάρτηση InitOpenGL() από το αρχείο systemWidow.cpp, που θέτει κάποιες default τιμές της OpenGL που έχουν σχέση με το περιεχόμενο του windows(π.χ το background και τα φώτα). Στην περίπτωση που η InitOpenGL() επιστρέψει FALSE, τότε το πρόγραμμα σταματάει να εκτελείται και επιστρέφει ένα αντίστοιχο μήνυμα σφάλματος. Σε αυτήν την περίπτωση το πιθανότερο είναι να χρειάζονται αναβάθμιση οι drivers της κάρτας γραφικών.

- Εκτελεί την συνάρτηση SoundPlay() από το αρχείο Scene.cpp. Στην περίπτωση που η SoundPlay() επιστρέψει FALSE, τότε το πρόγραμμα σταματάει να εκτελείται και επιστρέφει ένα αντίστοιχο μήνυμα σφάλματος στον χρήστη. Η συνάρτηση αυτή δημιουργεί ένα source, στην συνέχεια τοποθετεί τον buffer στον source και τον θετή σε Loop ON, πριν ξεκινήσει την εκτέλεσή του. Πρέπει να σημειωθεί, ότι χρησιμοποιήθηκαν extern μεταβλητές για την μεταφορά των buffers από το αρχείο LoadData.cpp στο Scene.cpp. Αν οποιαδήποτε από αυτές τις λειτουργίες επιστρέψει ένα σφάλμα, όλη η διαδικασία σταματάει και το πρόγραμμα κλείνει.

Πριν προχωρήσουμε στην επόμενη ενέργεια που πραγματοποιείται από το πρόγραμμα, θα αναλύσουμε λίγο την λειτουργία της LRESULT CALLBACK wndProc, που έχει άμεση σχέση με το τμήμα του κώδικα που ακολουθεί.

Η συγκεκριμένη συνάρτηση τοποθετείται πάντα πριν από την main και αποτελεί μια λειτουργία ανάκλησης της main. Είναι μια default συνάρτηση, που πρέπει να τοποθετείται πάντα, όταν δημιουργείται μια εφαρμογή που έχει άμεση σχέση με το λογισμικό των windows.

Ουσιαστικά η συνάρτηση αυτή συλλέγει όλα τα μηνύματα που στέλνονται από τα windows και εκτελεί τις ανάλογες λειτουργίες, σύμφωνα με όσα ορίζονται μέσα στην LRESULT CALLBACK wndProc. Ταυτόχρονα όμως, αν κάποια λειτουργία δεν έχει οριστεί, τότε θα εκτελεστεί η default διαδικασία, που ορίζουν τα windows για το παραθυρικό περιβάλον, θέτοντας ως return την DefWindowProc (hWnd, uMsg, wParam, lParam).

Επομένως μπορούμε να επικεντρώσουμε στο περιεχόμενο της συνάρτησης αυτής και να δούμε ορισμένες από τις λειτουργίες που ελέγχει.

```
bool  done=FALSE;        // Bool Variable To Exit Loop - Windows
bool  keys[256];         // Array Used For The Keyboard Routine
MSG         msg;         // Windows Message Structure
```

```
LRESULT CALLBACK WndProc(HWND hWnd, //Handle For This Window
                         UINT uMsg, //Message For This Window
                         WPARAM wParam,   //Additional Message Info
                         LPARAM lParam)   //Additional Message Info
{
```

Το τμήμα του κώδικα που ακολουθεί αποτρέπει την λειτουργία Screensaver και powersave. Επομένως ακόμα και αν απομακρυνθούμε για αρκετή ώρα από τον υπολογιστή και τα windows στείλουνε μήνυμα ότι πρέπει να ξεκινήσει μια από τις δύο λειτουργίες, καμία δεν θα πραγματοποιηθεί.

```
switch (uMsg)//Check For Windows Messages
{

case WM_SYSCOMMAND://Intercept System Commands
{
      switch (wParam){//Check System Calls

            case SC_SCREENSAVE://Screensaver Trying To Start
            case SC_MONITORPOWER://Monitor Trying To Enter Powersave.
            //By Returning 0 We Prevent Those Things From Happening.
            return 0;
      }
break;// Exit
}
```

Αν έχει προκύψει ένα μήνυμα εξόδου από την εφαρμογή, τότε στέλνουμε ένα μήνυμα στην main με την συγκεκριμένη πληροφορία και δεν επιτρέπουμε την εκτέλεση της διαδικασίας. Τα windows ορίζουν ότι, όταν μια εφαρμογή επεξεργάζεται αυτό το

μήνυμα, πρέπει να επιστρέψει μηδέν. Αυτό μας επιτρέπει να ρυθμίσουμε τον τρόπο που θα τερματιστεί η εφαρμογή, έχοντας προετοιμάσει και διαγράψει όλα τα αντικείμενα που δημιουργήσαμε κατά το ξεκίνημα της εφαρμογής.

```
//Sent As A Signal That A Window Or An Application Should Terminate.
case WM_CLOSE:
{
	PostQuitMessage(0);// Send A Quit Message
	return 0;		// Return To The Message Loop
}
```

Στην συνέχεια, χρησιμοποιώντας τον παρακάτω κώδικα, ελέγχουμε για μηνύματα των windows που αναφέρονται στο πάτημα ή την απελευθέρωση ενός κουμπιού του πληκτρολογίου και αποθηκεύουμε την παρακάτω τιμή στην μεταβλητή keys. Πρέπει να σημειωθεί ότι η συγκεκριμένη εντολή δεν αναφέρεται σε πλήκτρα συστήματος, όπως το prtcscn sysrq και το alt.

```
// The WM_KEYDOWN message is posted to the window with the keyboard
// focus when a nonsystem key is pressed. A nonsystem key is a key
// that is pressed when the ALT  key is not pressed.
//Info:http://msdn2.microsoft.com/en-us/library/ms646280.aspx
case WM_KEYDOWN:
{
// If a key is being held down we can find out what key it is by
// reading wParam. We then make that keys cell in the array keys[ ]
// become TRUE. That way I can read the array later on and find out
// which keys are being held down.This allows more than one key to be
// pressed at the same time.
	keys[wParam] = TRUE;// If So, Mark It As TRUE
	return 0;
}
//Info:http://msdn2.microsoft.com/en-us/library/ms646281.aspx
case WM_KEYUP:
{
//Each key on the keyboard can be represented by a number from 0-255.
//When I press the key that represents the number 40 for example,
//keys[40] will become TRUE. When I let go, it will become FALSE.
	keys[wParam] = FALSE;
	return 0;
}
```

Σε αυτό το σημείο μπορούμε να επιστρέψουμε στην main και να εξετάσουμε τον κώδικα που ακολουθεί μετά την WindowsRAN:

```
while(!done)
{
	//Is There A Message Waiting?
	if (PeekMessage(&msg,NULL,0,0,PM_REMOVE)){
	//Have We Received A Quit Message?
		if (msg.message==WM_QUIT){
			done=TRUE;  //If So done=TRUE
		}
		else{ //If Not, Deal With Window Messages
			TranslateMessage(&msg); //Translate The Message
			DispatchMessage(&msg);  //Dispatch The Message
		}
	}
	else{//If There Are No Messages
```

Όπως φαίνεται, το πρώτο πράγμα που κάνουμε είναι να ξεκινήσουμε μια ρουτίνα που θα τερματιστεί μόνο αν η bool done γίνει TRUE. Υπενθυμίζουμε ότι στην αρχή του προγράμματος είχαμε θέσει αυτήν την μεταβλητή σε FALSE.

Στην συνέχεια ελέγχουμε τα μηνύματα που έχουν αποσταλεί από την LRESULT CALLBACK wndProc και ειδικότερα αν έχει σταλεί μήνυμα εξόδου από το πρόγραμμα. Σε αυτήν την περίπτωση τερματίζουμε την ρουτίνα που δημιουργήσαμε και εκτελούμε τον κώδικα που επιθυμούμε, ώστε να κλείσουμε το πρόγραμμα. Αν δεν υπάρχει κάποιο μήνυμα εξόδου, συνεχίζουμε εξετάζοντας τα υπόλοιπα μηνύματα που αποστέλλονται στην LRESULT CALLBACK wndProc και ειδικότερα αν έχει πατηθεί το πλήκτρο Esc. Αν και το συγκεκριμένο πλήκτρο δεν έχει πατηθεί, εκτελούμε τον υπόλοιπο κώδικα.

```
if (active && keys[VK_ESCAPE]){//  Was There A Quit Received?
      done=TRUE;//Quit
}
else{//Star DrawGLScene
// Not Time To Quit, Update Screen
      DrawGLScene();// Draw The Scene
```

Αρχικά εκτελούμε την συνάρτηση DrawGLScen. Η συνάρτηση αυτή περιέχει οτιδήποτε αλλάζει μέσα στο παράθυρο, όσον αφορά τους ήχους και το interface. Θα αναλυθεί αφού ολοκληρωθεί η ανάλυση της ρουτίνας. Επομένως όταν εκτελεστεί η συνάρτηση, ελέγχουμε για το πάτημα των πλήκτρων Page up –down καθώς και τα βελάκια του πληκτρολογίου. Στην περίπτωση που η LRESULT CALLBACK wndProc μας έχει δώσει το μήνυμα ότι κάποιο από τα συγκεκριμένα πλήκτρα έχουν πατηθεί, τότε αυξάνουμε ή μειώνουμε την αντίστοιχη τιμή.

```
if (keys[VK_PRIOR]){// Is Page Up Being Pressed
      PosUD+=0.003f;// If So, Move UP
      if(PosUD>0.771001f){ //if Max up
            PosUD = 0.771001f;
      }
      else{
            ypos+=0.03f;
      }
}
if (keys[VK_NEXT]){// Is Page Down Being Pressed
      PosUD-=0.003f;// If So, Move Down
      if(PosUD<-0.501000f ){ //if Max Down
            PosUD = -0.501000f;
      }
      else{
            ypos-=0.03f;
      }
}
if (keys[VK_UP]){ // Is Up Arrow Being Pressed
      PosIO+=0.003f; // If So, Move Into The Screen
      if(PosIO>0.771001f){//if Max up
            PosIO = 0.771001f;
      }
      else{
            zpos-=0.03f;
      }
}
```

```
if (keys[VK_DOWN]){ // Is Down Arrow Being Pressed
      PosIO-=0.003f;// If So, Move Towards The Viewer
      if(PosIO<-0.501000f){   //if Max Down
            PosIO = -0.501000f;
      }
      else{
            zpos+=0.03f;
      }
}
if (keys[VK_RIGHT]){// Is Right Arrow Being Pressed
      PosLR+= 0.003f;// If So, Move Right
      if(PosLR>0.771001f){ //if Max Right
            PosLR = 0.771001f;
      }
      else {
            xpos+=  0.03f;
      }
}

if (keys[VK_LEFT]){// Is Left Arrow Being Pressed
      PosLR-= 0.003f;// If So, Move Left
      if(PosLR<-0.771001f){ //if Max Left
            PosLR = -0.771001f;
      }
      else {
            xpos-=  0.03f;
      }
}
```

Οι τιμές αυτές για κάθε πλήκτρο προσδιορίζουν δύο θέσεις και δύο διαφορετικές μετακινήσεις.

Η μια μετακίνηση είναι ηχητική και αναφέρεται στα x,y,z του source που έχουν δηλωθεί στην αρχή του προγράμματος ως εξής:

```
GLfloat ypos;                        //Position Up -Down
GLfloat zpos;                        //Position Inside - Outside
GLfloat xpos;                        //Position Right -Left
```

Η άλλη μετακίνηση αφορά το γραφικό interface και ουσιαστικα τα τρία button που μετακινούνται και έχουν δηλωθεί στην αρχή του προγράμματος ως εξής:

```
GLfloat PosUD  = 0.135f;      //Position Up -Down
GLfloat PosIO  = 0.135f;      //Position Inside - Outsude
GLfloat PosLR;                //Position Right -Left
```

Αυτό που πρέπει να γίνει πλήρως κατανοητό από τον παραπάνω κώδικα είναι ότι, η LRESULT CALLBACK wndProc στέλνει πληροφορίες καθ όλη την διάρκεια που εκτελείται το πρόγραμμα και ταυτόχρονα, μέσω της ρουτίνας που δημιουργήσαμε, ερμηνεύουμε συνέχεια τις τιμές που μας ενδιαφέρουν σε πραγματικό χρόνο, ενώ εκτελούμε την συνάρτηση DrawGLScen, που αποτελεί τον πυρήνα όλου του παραδείγματος και την ερμηνεία όλων αυτών των τιμών.

Επομένως μεταφέρουμε τις τιμές στο αρχείο Scene.cpp :

```
//extern form Main.cpp
extern GLfloat  PosUD;          //Position Up -Down
extern GLfloat  PosIO;         //Position Inside - Outsude
extern GLfloat  PosLR;         //Position Right -Left
//sound
extern GLfloat    zpos;        //Position Inside - Outside
extern GLfloat    ypos;        //Position Up -Down
extern GLfloat    xpos;        //Position Right -Left
```

Υπενθυμίζουμε ότι μεταφέρουμε και τις εξής τιμές από το αρχείο LoadData.cpp:

```
//extern form LoadData.cpp
extern GLuint   texture[6]; //load texture data
extern ALuint   Buffer [1];   //load Sound data
```

Κατά την διάρκεια της εκκίνησης του προγράμματος έχουμε τρέξει την συνάρτηση SoundPlay() και, επομένως, πριν ξεκινήσουμε την ρουτίνα, ο ήχος ήδη έχει τοποθετηθεί στον source και εκτελείται.

```
ALuint   Sources[1];

ALboolean SoundPlay(){

      ALenum      Error;

      alGetError ();//Clear AL error code

      //Generate 4 Source
      alGenSources (1,Sources);
      Error = alGetError ();
      if ( Error != AL_NO_ERROR )
            return AL_FALSE;

      //Attach Buffer to Sources
      alSourcei (Sources[0], AL_BUFFER, Buffer[0]);
      Error = alGetError ();
      if ( Error != AL_NO_ERROR )
            return AL_FALSE;

      //set sound Loop ON
      alSourcei(Sources[0], AL_LOOPING, AL_TRUE);//Loop ON
      if ( Error != AL_NO_ERROR )
            return AL_FALSE;

      //play a  source
      alSourcePlay (Sources[0]);
      Error = alGetError ();
      if ( Error != AL_NO_ERROR )
            return AL_FALSE;



return AL_TRUE;
}
```

Κάθε φορά που εκτελείται η συνάρτηση DrawGLScen, εκκαθαρίζονται οι buffers της GL και στην συνέχεια τοποθετείται το Matrix στην αρχική του θέση, όπου είναι το κέντρο της οθόνης.

```
GLboolean DrawGLScene(){//All  Drawing System

GLboolean         bReturn = GL_TRUE;


// Clear The Screen And The Depth Buffer
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
//Replace The Current Matrix With The Identity Matrix
glLoadIdentity();
```

Αρχικά θέτουμε τις τιμές που παίρνουμε από την ρουτίνα στον source.

```
      ALfloat Spos1[] = {xpos,ypos,zpos}; //Position of the listener
      alSourcefv(Sources[0], AL_POSITION, Spos1); //Move Source
```

Πρέπει να σημειωθεί ότι η GL έχει την ίδια φιλοσοφία, όσον αφορά την κίνηση αντικειμένων στον εικονικό κόσμο, με την AL και επομένως αν θέλαμε να μετακινήσουμε ελευθέρα ένα αντικείμενο ηχητικά και γραφικά στις τρεις διαστάσεις, θα τοποθετούσαμε τις ίδιες τιμές και στην AL και στην GL. Στο συγκεκριμένο παράδειγμα όμως μια ηχητική μετακίνηση διαμορφώνεται από την κίνηση τριών διαφορετικών γραφικών αντικειμένων (controllers), που μετακινούνται ή πάνω-κάτω ή δεξιά-αριστερά. Για αυτόν τον προφανή λόγο χρησιμοποιούνται τα PosUD, PosIO,PosLR.

Επομένως χρησιμοποιούμε τις εντολές push και pop matrix, που αναγκάζουν το matrix να εκτελέσει τις εντολές που περιέχονται μέσα σε αυτές και στην συνεχεία να επανέλθει στην προηγούμενη κατάσταση του, που ήταν το σημείο που έβλεπε ο χρήστης δηλαδή το σημείο (0.0,0.0,0.0). Έτσι τοποθετούμε το matrix στο σημείο που έχουμε ορίσει ότι θα δημιουργηθεί το background (0.0f,0.0f,-4.0f), μέσω της εντολής glTranslatef. Στην συνεχεία δηλώνουμε ότι θα χρησιμοποιήσουμε το αντίστοιχο texture και εντέλει σχεδιάζουμε ένα τετράγωνο.

```
//back
glPushMatrix();
glTranslatef( 0.0f,0.0f, -4.0f);//Move Into The Screen
glBindTexture(GL_TEXTURE_2D, texture[filter]);//Get tetxure
      glBegin(GL_QUADS);
      glNormal3f( 0.0f, 0.0f, 1.0f);// Normal Pointing Down
      glTexCoord2f(0.0f, 0.0f); glVertex3f(-1.0f, -1.0f,  1.0f);
      glTexCoord2f(1.0f, 0.0f); glVertex3f( 1.0f, -1.0f,  1.0f);
      glTexCoord2f(1.0f, 1.0f); glVertex3f( 1.0f,  1.0f,  1.0f);
      glTexCoord2f(0.0f, 1.0f); glVertex3f(-1.0f,  1.0f,  1.0f);
      glEnd();
glPopMatrix();
```

Σε αυτό το σημείο θα αναλύσουμε τον τρόπο δημιουργίας του τετραγώνου. Η συνάρτηση normal δίνει την τοποθεσία του «λογικού» σημείου που θα τοποθετηθεί εντέλει το τετράγωνο που θα δημιουργήσουμε, ώστε να δουλέψει σωστά ο φωτισμός. Επομένως η πραγματική θέση του τετραγώνου θα είναι Z =2.0F και όχι 1.0f.

Στην συνέχεια, εκτελούμε δύο εντολές για κάθε ένα από τα τέσσερα σημεία που ουσιαστικά δημιουργούμε, τα οποία ενώνονται και δημιουργούν το τετράγωνο. Η πρώτη συνάρτηση αναφέρεται στο πως θα τοποθετηθεί το texture, ενώ η δεύτερη αναφέρεται στα τέσσερα σημεία του τετραγώνου.

Η τιμή (-1.0f, -1.0f, 1.0f) σημαίνει μια μετατόπιση στον άξονα x κατά -1, δηλαδή αριστερά, μια μετατόπιση στον άξονα y κατά -1, δηλαδή κάτω και μια μετατόπιση στον άξονα z προς τα έξω. Όπως φαίνεται το σημείο αυτό θα είναι το κάτω αριστερά. Το επόμενο σημείο (1.0f, -1.0f, 1.0f) θα είναι το κάτω δεξιά. Όλα τα σημεία αυτά έχουνε ως κεντρικό άξονα το σημείο που ορίσαμε με την glTranslatef, δηλαδή την πρώτη φορά που μετακινείται κατά (-1.0f, -1.0f, 1.0f) πραγματικά μεταφέρεται στο σημείο (-1.0f, -1.0f,-3.0f).

Εντέλει με τον ίδιο τρόπο τοποθετούμε όλα τα σημεία του τετραγώνου και συνεχίζουμε με την δημιουργία του πρώτου κινούμενου αντικειμένου –«κουμπιού».

```
//batton Up Down
glPushMatrix();
glBindTexture(GL_TEXTURE_2D, texture[button]);//Get tetxure
//Move Into The Screen. Normal-Zero is y=0.135
glTranslatef( -0.63f, PosUD,-3.0f);
      glBegin(GL_QUADS);
      glNormal3f( 0.0f, 0.0f, 0.05f);// Normal Pointing Down
      glTexCoord2f(0.0f, 0.0f); glVertex3f(-0.05f, -0.015f,  0.05f);
      glTexCoord2f(1.0f, 0.0f); glVertex3f( 0.05f, -0.015f,  0.05f);
      glTexCoord2f(1.0f, 1.0f); glVertex3f( 0.05f,  0.015f,  0.05f);
      glTexCoord2f(0.0f, 1.0f); glVertex3f(-0.05f,  0.015f,  0.05f);
      glEnd();
glPopMatrix();
```

Όπως βλέπουμε στην εντολή glTranslatef θέτουμε το PosUD, ώστε να κινείται το αντικείμενο κάθε φορά που επιρεάζεται ο controller up–Down. Στην συνέχεια σχεδιάζουμε ένα πολύ μικρό ορθογώνιο που είναι τοποθετημένο μπροστά από το background που δημιουργήσαμε. Η κίνηση του αντικειμένου αυτού δημιουργείται από τον παρακάτω κώδικα, που ήδη έχουμε δει:

```
if (keys[VK_PRIOR]){           // Is Page Up Being Pressed
     PosUD+=0.003f;            // If So, Move UP
     ypos+=0.03f;
     if(PosUD>0.771001f){ //if Max up
          PosUD = 0.771001f;
          ypos  = 6.360013f;
      }
}
if (keys[VK_NEXT]){            // Is Page Down Being Pressed
      PosUD-=0.003f;
      ypos-=0.03f;            // If So, Move Down
      if(PosUD<-0.501000f ){ //if Max Down
            PosUD = -0.501000f;
            ypos  = -6.360013f;
      }
}
```

Όπως φαίνεται κάθε φορά που πατάμε ένα πλήκτρο αυξομειώνουμε αντίστοιχα και την θέση του Source στον άξονα y, αλλά και την θέση του αντικειμένου, επίσης στον άξονα y. Όταν εντέλει φτάσουμε σε μέγιστο ή ελάχιστο γραφικό σημείο (0.771001f ή 0.501000f), σταματάμε την μετατόπιση του αντικειμένου, αλλά και του ήχου, δημιουργώντας έτσι ένα Graphical user interface.

Η αντίστοιχη διαδικασία εκτελείται και για τα υπόλοιπα δύο κουμπιά, όπως παρουσιάζεται στον παρακάτω κώδικα :

```
//batton Inside - Outside
glPushMatrix();
	glBindTexture(GL_TEXTURE_2D, texture[button]);//Get tetxure
	//Move Into The Screen. Normal-Zero is y=0.135
	glTranslatef( 0.63f,PosIO,-3.0f);
	glBegin(GL_QUADS);
	glNormal3f( 0.0f, 0.0f, 0.05f);// Normal Pointing Down
	glTexCoord2f(0.0f, 0.0f); glVertex3f(-0.05f, -0.015f,  0.05f);
	glTexCoord2f(1.0f, 0.0f); glVertex3f( 0.05f, -0.015f,  0.05f);
	glTexCoord2f(1.0f, 1.0f); glVertex3f( 0.05f,  0.015f,  0.05f);
	glTexCoord2f(0.0f, 1.0f); glVertex3f(-0.05f,  0.015f,  0.05f);
	glEnd();
glPopMatrix();

//batton Right -Left
glPushMatrix();
	glBindTexture(GL_TEXTURE_2D, texture[button]);//Get tetxure
	//Move Into The Screen. Normal-Zero is X=0.0
	glTranslatef( PosLR,-0.86f,-3.0f);
	glBegin(GL_QUADS);
	glNormal3f( 0.0f, 0.0f, 0.05f);// Normal Pointing Down
	glTexCoord2f(0.0f, 0.0f); glVertex3f(-0.015f, -0.05f,  0.05f);
	glTexCoord2f(1.0f, 0.0f); glVertex3f( 0.015f, -0.05f,  0.05f);
	glTexCoord2f(1.0f, 1.0f); glVertex3f( 0.015f,  0.05f,  0.05f);
	glTexCoord2f(0.0f, 1.0f); glVertex3f(-0.015f,  0.05f,  0.05f);
	glEnd();
glPopMatrix();

	return bReturn;
}
```

Εντέλει, αφού ολοκληρωθούν όλες αυτές οι διαδικασίες, το πρόγραμμα επιστρέφει στην ρουτίνα done και την επανεκτελεί, μέχρις ότου να λάβει ένα μήνυμα εξόδου. Όταν πραγματοποιηθεί η έξοδος από την ρουτίνα, εκτελούνται τέσσερις συναρτήσεις πριν να κλείσει το πρόγραμμα.

```
/* ================================================================
/*                      Exit -Close System
/* ================================================================
	ExitPlay();
	CloseALUT();//Exit-Close ALUT
	CloseOpenAL();//Exit-Close OpenAL
	ExitGLWindow();//Close Window -OpenAL
return (msg.wParam);//Exit The Program.Return PostQuitMessage(0);
}
```

Η συνάρτηση ExitGLWindows κλείνει την OpenGL και το δημιουργημένο παράθυρο. Οι συναρτήσεις CloseALUT και CloseAL κλείνουν τα αντίστοιχα API, όπως έχει περιγραφεί στα προηγούμενα παραδείγματα. Η συνάρτηση ExitPlay αποδεσμεύει και κλείνει όλα τα Objects που δημιουργήθηκαν με την SoundPlay.

Ο κώδικας της συγκεκριμένης συνάρτησης φαίνεται παρακάτω:

```
ALvoid ExitPlay(){

      alSourcef(Sources[0],AL_GAIN,0.1f);
      alutSleep(1);
      alSourceStop(Sources[0]);
      alDeleteSources(1,Sources);
      alDeleteBuffers(1,Buffer);
}
```

Ολοκληρώνοντας το όγδοο παράδειγμα της πτυχιακής εργασίας παρατίθεται ο πλήρης κώδικας του συγκεκριμένου παραδείγματος:

```
                                    /*Include.h*/
#ifndef INCLUDE_H_
#define INCLUDE_H_
/* ================================================================ */
/*                       Inlcude  Libraries                         */
/* ================================================================ */

//#pragma comment(lib, "winmm.lib") //Link With Windows MultiMedia
//lib Link With OpenAL Lib
#pragma comment(lib, "OpenAL32.lib")
#pragma comment(lib, "alut.lib")
#pragma comment(lib, "EFX-Util.lib")

//Link With Microsoft OpenGL lib
#pragma comment (lib, "opengl32.lib")
//Link With Microsoft OpenGL Utility lib
#pragma comment (lib, "glu32.lib")
//Link With Win32 GLAUX lib
#pragma comment (lib, "glaux.lib")
//#pragma comment (lib, "glut32.lib")//Link With GLUT  lib

/* ================================================================ */
/*                Standard Include Libraries Files                  */
/*================================================================== */
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <stdarg.h>
#include <Windows.h>
#include <ctype.h>
#include <math.h>
#include <time.h>
#include <conio.h>
#include <malloc.h>
#include <memory.h>
#include <tchar.h>
#include <iostream>
```

```
/* ================================================================ */
/*                Reporting Errors Standard Libraries Files         */
/* ================================================================ */
#include <errno.h> //int errno
#include <assert.h>//assert(Value);

/* ================================================================ */
/*    Inlcude Full Sound OpenlAL Libraries Files                    */
/* ================================================================ */
#include <al.h>
#include <alc.h>
#include <alut.h>
#include <efx.h>

/* ================================================================ */
/*          Include Full Graphic OpenGL Libraries Files             */
/* ================================================================ */
#include <gl/GL.h>//VC LIB
#include <gl/GLU.h>//VC LIB
#include <gl/GLAux.h>//VC LIB
//#include <gl/GLUT.h>

/* ================================================================ */
/*                Inlcude Program  Libraries    Files               */
/* ========== ================================================== */
#include "SystemWindow.h"  //Window Setup
#include "InitExitAL.h"    //Initializing & exiting OpenAL and ALUT
#include "LoadData.h"      //Load Data
#include "Scene.h"         //Window Scene

/* ================================================================ */
/*                 CALLBACK -Declaration For WndProc                */
/* ================================================================ */
LRESULT     CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

#endif /*MAININCLUDE_H_*/
```

/*InitExitAL.h */

```
#ifndef INITEXITAL_H_
#define INITEXITAL_H_


/*================================================================*/
/*                Initialization & Exit Sound Function            */
/* ================================================================*/
//AL Sound Full Setup (If No Device Sound Close)*/
ALboolean ALSound ();
ALboolean InitOpenAL();//Initialization OpenAL
ALboolean CloseOpenAL();//Exit-Close OpenAL
ALboolean InitALUT();//Initialization ALUT
ALboolean CloseALUT();//Exit-Close ALUT

#endif /*INITEXITAL_H_*/
```

/*InitExitAL.cpp*/

```
#include "Include.h"
/* ================================================================= */
/*                Initialization & Exit Sound Function               */
/* ================================================================= */

ALboolean ALSound (){//AL Sound Full Setup (If No Device Sound Close)

      if (InitOpenAL() == AL_FALSE ){//Initialization OpenAL
            CloseOpenAL();
            return AL_FALSE;
      }
      else if(InitALUT() == AL_FALSE ){//Initialization ALUT
          CloseALUT();
          CloseOpenAL();
            return AL_FALSE;
      }

      return AL_TRUE;
}
```

```
ALboolean InitOpenAL()//Initialization OpenAL Manually
{
ALCcontext        *pContext = NULL;
ALCdevice         *pDevice = NULL;
ALboolean         bReturn = AL_FALSE;
ALCboolean        currentcon ;//Test For alcMakeContextCurrent

      pDevice = alcOpenDevice(NULL); //Open default device
      if (pDevice){ //pDevice != NULL
            //creates a context using a default device
            pContext = alcCreateContext(pDevice, NULL);
            if (pContext){//pContext != NULL
            //set active context-Makes a pContext the current context
                  currentcon = alcMakeContextCurrent(pContext);
                  if (currentcon == ALC_TRUE){
                        bReturn = AL_TRUE;
                  }
            }
      }

      return bReturn;
}
```

```
ALboolean CloseOpenAL()//Exit-Close OpenAL
{
ALCcontext              *pContext = NULL;
ALCdevice               *pDevice  = NULL;

pContext = alcGetCurrentContext(); //open the current context
//open a context's device pointer
pDevice = alcGetContextsDevice(pContext);
alcMakeContextCurrent(NULL);//makes NULL context the current context
alcDestroyContext(pContext);//destroys a context
alcCloseDevice(pDevice);//closes a device

      return AL_TRUE;
}
```

```
ALboolean  InitALUT()//Initialization ALUT
{
      ALboolean               bReturn = AL_TRUE;


      if (!alutInitWithoutContext( NULL , NULL ))
                  bReturn = AL_FALSE;

      return bReturn;
}
```

```
ALboolean CloseALUT()//Exit-Close ALUT
{
      ALboolean               bReturn = AL_TRUE;

      if (!alutExit())
            bReturn = AL_FALSE;

      return bReturn;
}
```

/*LoadData.h*/

```
#ifndef LOADDATA_H_
#define LOADDATA_H_
/* ================================================================ */
/*                       General Function                           */
/*================================================================ */
//Path Finder.Test if Path Exist,If Not return FALSE.
//Sound   Data Path is :SoundPath    : Data\\Sounds\\filename
//Texture Data Path is :TexturesPath : Data\\Textures\\filename
char *DataPath(const char *Path,const char *filename);
/* ================================================================ */
/*    Load GLTextures Data - Definition Variable                    */
/* ================================================================  */
//Texture Path :....Data/Textures/
#define TexturesPath          "Data\\Textures\\"
#define texrure1              "back.bmp"
#define texrure2              "buton.bmp"/*
================================================================ */
/*                 Load GLTextures Data Function                   */
/* ================================================================ */
//Loads A Bitmap Image.If Load Failed Return FALSE
AUX_RGBImageRec *LoadBMP(const char *Filename);
// Load Bitmaps And Convert To Textures
GLuint LoadGLTextures();
/* ================================================================ */
/*           Load Sound Data - Definition Variable                 */
/* ================================================================ */
//Sound Path :....Data/Sounds/
#define SoundPath             "Data\\Sounds\\"
#define Sound1                "1.WAV"

/* ================================================================ */
/*          Load Sound Data Function                               */
/* ================================================================ */
//Load a .WAV sound Be Name.If Load Failed Return AL_FALSE
ALuint      LoadWavSound();

#endif /*LOADDATA_H_*/
```

/*LoadData.cpp*/

```
#include "Include.h"
/* ============================================================= */
/*          General Function                                     */
/* ============================================================= */
char fullPath[MAX_PATH];
char *DataPath(const char *Path,const char *filename){
//Path Finder.Test if Path Exist,If Not return FALSE.
//Sound   Data Path is :SoundPath    : Data\\Sounds\\filename
//Texture Data Path is :TexturesPath : Data\\Textures\\filename

      FILE *File=NULL;// File Handle
      if (!filename||!Path){//Make Sure A Filename and Path Was Given
            return FALSE;//If Not Return FALSE
      }
     //Make spring of full Path
      sprintf_s(fullPath,MAX_PATH,"%s%s",Path,filename);
     //Check If The  full Path File Exists-File=fopen(fullPath,"r");
      fopen_s(&File,fullPath,"r");

      if (File){// IF File Exist?
            fclose(File);//Close The Handle
            return fullPath;
      }
      return FALSE;//If Not Return FALSE
}
```

```
/* ============================================================= */
/*               Load GLTextures Data Function                   */
/* ============================================================= */
// RGB IMAGE INFO:The image height and width MUST be a power of 2.
// The width and height must be at least 64 pixels, and for
// compatability reasons, shouldn't be more than 256 pixels. If the
// image you want to use is not 64, 128 or 256 pixels on the width or
// height, resize it in an art program.
//AUX_RGBImageRec : The record will hold the bitmap width,height,and
//data

AUX_RGBImageRec *LoadBMP(const char *Filename)
{//Loads A Bitmap Image.If Load Failed Return FALSE
      char  *fullPath;//Full Path Texture file

      fullPath = DataPath(TexturesPath,Filename);//Path Finder

      if (fullPath){//If full Path exist
            //Load The Bitmap And Return A Pointer
            return auxDIBImageLoad(fullPath);
      }
      return FALSE;//If Not Return NULL
}

GLuint      texture[6];

GLuint LoadGLTextures()
{// Load Bitmaps And Convert To Textures

int               loop;
int               bReturn = GL_FALSE;
GLenum            ErrorGL;//Save GL Error - GL_NO_ERROR.
```

```
glGetError();//Clear error code


// Create Storage Space For The Texture-Create an image record that
// we can store our bitmap in.
// The record will hold the bitmap width, height, and data.
AUX_RGBImageRec *TextureImage[2];

// Clear the image record just to make sure it's empty.
// Set The Pointer To NULL
// memset Info:
// http://www.cplusplus.com/reference/clibrary/cstring/memset.html
// Sizeof Info:
// http://msdn2.microsoft.com/en-us/library/4s7x1k91(VS.71).aspx
// void is not a type of data in this usage, but indicates the
// absence of data.A void* points at objects of unknown size,
// so pointer arithmetic is not defined on them. In C++ the use
// of 'void*' for addresses of objects of an unknown type is still
// legal.http://www.csci.csusb.edu/dick/samples/c++.glossary.html
memset(TextureImage,0,sizeof(void *)*2);

// Load The Bitmap
TextureImage[0]=LoadBMP(texrure1);
TextureImage[1]=LoadBMP(texrure2);

//Check For Errors, If Bitmap's Not Found Quit.
if (TextureImage[0] && TextureImage[1])
{
bReturn = GL_TRUE;// Set The Status To TRUE
glGenTextures(6, &texture[0]);// Create 4 Texture

//////////////////back////////////////////////////////////////
//Low
// Create Nearest Filtered TextureThis type of texture has
// no filtering at all.The only benefit of this type of texture is
that
// projects made using this type of texture will usually run pretty
// good on slow computers.
glBindTexture(GL_TEXTURE_2D, texture[0]);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_NEAREST);
glTexImage2D(GL_TEXTURE_2D, 0, 3, TextureImage[0]->sizeX,
             TextureImage[0]->sizeY, 0, GL_RGB, GL_UNSIGNED_BYTE,
             TextureImage[0]->data);
//Mid
// Create Linear Filtered Texture. Using GL_LINEAR
// requires alot of work from the processor/video card.
glBindTexture(GL_TEXTURE_2D, texture[1]);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_LINEAR);
glTexImage2D(GL_TEXTURE_2D, 0, 3, TextureImage[0]->sizeX,
             TextureImage[0]->sizeY, 0, GL_RGB, GL_UNSIGNED_BYTE,
             TextureImage[0]->data);

//High
// Create MipMapped Texture
// OpenGL to build a mipmapped texture OpenGL tries to build
// different sized high quality textures. When you draw a mipmapped
// texture to the screen OpenGL will select the BEST looking texture
```

```
// from the ones it built (texture with the most detail) and draw
// it to the screen instead of resizing the original image (which
// causes detail loss).
glBindTexture(GL_TEXTURE_2D, texture[2]);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,
                GL_LINEAR_MIPMAP_NEAREST);
gluBuild2DMipmaps(GL_TEXTURE_2D, 3, TextureImage[0]->sizeX,
                  TextureImage[0]->sizeY, GL_RGB, GL_UNSIGNED_BYTE,
                  TextureImage[0]->data);

////////////////////////////////button//////////////////////////////////
//Low
glBindTexture(GL_TEXTURE_2D, texture[3]);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_NEAREST);
glTexImage2D(GL_TEXTURE_2D, 0, 3, TextureImage[1]->sizeX,
              TextureImage[1]->sizeY, 0, GL_RGB, GL_UNSIGNED_BYTE,
              TextureImage[1]->data);

//Mid
glBindTexture(GL_TEXTURE_2D, texture[4]);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_LINEAR);
glTexImage2D(GL_TEXTURE_2D, 0, 3, TextureImage[1]->sizeX,
             TextureImage[1]->sizeY, 0, GL_RGB, GL_UNSIGNED_BYTE,
             TextureImage[1]->data);

//High
glBindTexture(GL_TEXTURE_2D, texture[5]);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,
                GL_LINEAR_MIPMAP_NEAREST);
gluBuild2DMipmaps(GL_TEXTURE_2D, 3, TextureImage[1]->sizeX,
                  TextureImage[1]->sizeY, GL_RGB, GL_UNSIGNED_BYTE,
                  TextureImage[1]->data);

ErrorGL=glGetError();//!!!Error ????
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE; //IF Error Set The Status To FALSE

for (loop=0; loop<2; loop++)  // Loop Through 4 Textures
{
      if (TextureImage[loop]) // If Texture Exists
      {
            if (TextureImage[loop]->data) // If Texture Image Exists
            {
                  // Free The Texture Image Memory
                  free(TextureImage[loop]->data);
            }
                 // Free The Image Structure
                  free(TextureImage[loop]);
                  }
      }
}

return bReturn;
}
```

```
/* ================================================================ */
/*                 Load Sound Data Function                         */
/* ================================================================ */

ALuint Buffer[1];

ALuint  LoadWavSound ()
{ //Load a .WAV sound Be Name.If Load Failed Return AL_FALSE

      ALchar         *fullPath;

      fullPath  = DataPath(SoundPath,Sound1);
      Buffer[0] = alutCreateBufferFromFile ( fullPath);


      //Check For Errors, If Sound's Not Found Quit.
      if (Buffer[0]== AL_NONE)
      {
            alDeleteBuffers(1,Buffer);//delete Buffers
            return AL_FALSE;
      }

      return AL_TRUE;
}
```

/*Scene.h*/

```
#ifndef SCENE_H_
#define SCENE_H_

//All Drawing System
GLboolean DrawGLScene();
//sound sutup and paly
ALboolean SoundPlay();
//Exit Sound
ALvoid ExitPlay();

#endif /*SCENE_H_*
```

/*Scene.cpp*/

```
#include "Include.h"


ALuint   Sources[1];
//extern form LoadData.cpp
extern GLuint   texture[6];   //load texture data
extern ALuint   Buffer [1];   //load Sound data


//extern form Main.cpp
extern GLuint   filter;       // Which Filter To Use
extern GLfloat  PosUD;        //Position Up -Down
extern GLfloat  PosIO;        //Position Inside - Outsude
extern GLfloat  PosLR;        //Position Right -Left
//sound
extern GLfloat  zpos;         //Position Inside - Outside
extern GLfloat  ypos;         //Position Up -Down
extern GLfloat  xpos;         //Position Right -Left
```

```
GLboolean DrawGLScene(){//All  Drawing System

GLboolean          bReturn = GL_TRUE;
GLuint button      =filter+3;//button filter

// Clear The Screen And The Depth Buffer
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
//Replace The Current Matrix With The Identity Matrix
glLoadIdentity();

ALfloat Spos1[] = {xpos,ypos,zpos}; //Position of the listener
alSourcefv(Sources[0], AL_POSITION, Spos1); //Move Source

//back
glPushMatrix();
      glBindTexture(GL_TEXTURE_2D, texture[filter]);//Get tetxure
      glTranslatef( 0.0f,0.0f, -4.0f);//Move Into The Screen
      glBegin(GL_QUADS);
      glNormal3f( 0.0f, 0.0f, 1.0f);// Normal Pointing Down
      glTexCoord2f(0.0f, 0.0f); glVertex3f(-1.0f, -1.0f,  1.0f);
      glTexCoord2f(1.0f, 0.0f); glVertex3f( 1.0f, -1.0f,  1.0f);
      glTexCoord2f(1.0f, 1.0f); glVertex3f( 1.0f,  1.0f,  1.0f);
      glTexCoord2f(0.0f, 1.0f); glVertex3f(-1.0f,  1.0f,  1.0f);
      glEnd();
glPopMatrix();
//batton Up Down
glPushMatrix();
      glBindTexture(GL_TEXTURE_2D, texture[button]);//Get tetxure
      //Move Into The Screen. Normal-Zero is y=0.135
      glTranslatef( -0.63f, PosUD,-3.0f);
      glBegin(GL_QUADS);
      glNormal3f( 0.0f, 0.0f, 0.05f);// Normal Pointing Down
      glTexCoord2f(0.0f, 0.0f); glVertex3f(-0.05f, -0.015f,  0.05f);
      glTexCoord2f(1.0f, 0.0f); glVertex3f( 0.05f, -0.015f,  0.05f);
      glTexCoord2f(1.0f, 1.0f); glVertex3f( 0.05f,  0.015f,  0.05f);
      glTexCoord2f(0.0f, 1.0f); glVertex3f(-0.05f,  0.015f,  0.05f);
      glEnd();
glPopMatrix();

//batton Inside - Outside
glPushMatrix();
      glBindTexture(GL_TEXTURE_2D, texture[button]);//Get tetxure
      //Move Into The Screen. Normal-Zero is y=0.135
      glTranslatef( 0.63f,PosIO,-3.0f);
      glBegin(GL_QUADS);
      glNormal3f( 0.0f, 0.0f, 0.05f);// Normal Pointing Down
      glTexCoord2f(0.0f, 0.0f); glVertex3f(-0.05f, -0.015f,  0.05f);
      glTexCoord2f(1.0f, 0.0f); glVertex3f( 0.05f, -0.015f,  0.05f);
      glTexCoord2f(1.0f, 1.0f); glVertex3f( 0.05f,  0.015f,  0.05f);
      glTexCoord2f(0.0f, 1.0f); glVertex3f(-0.05f,  0.015f,  0.05f);
      glEnd();
glPopMatrix();

//batton Right -Left
glPushMatrix();
      glBindTexture(GL_TEXTURE_2D, texture[button]);//Get tetxure
      //Move Into The Screen. Normal-Zero is X=0.0
      glTranslatef( PosLR,-0.86f,-3.0f);
      glBegin(GL_QUADS);
```

```
	glNormal3f( 0.0f, 0.0f, 0.05f);// Normal Pointing Down
	glTexCoord2f(0.0f, 0.0f); glVertex3f(-0.015f, -0.05f,  0.05f);
	glTexCoord2f(1.0f, 0.0f); glVertex3f( 0.015f, -0.05f,  0.05f);
	glTexCoord2f(1.0f, 1.0f); glVertex3f( 0.015f,  0.05f,  0.05f);
	glTexCoord2f(0.0f, 1.0f); glVertex3f(-0.015f,  0.05f,  0.05f);
	glEnd();
glPopMatrix();

	return bReturn;
}
```

```
ALboolean SoundPlay(){

ALenum      Error;

alGetError ();//Clear AL error code

//Generate 1 Source
alGenSources (1,Sources);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
	return AL_FALSE;

//Attach Buffer to Sources
alSourcei (Sources[0], AL_BUFFER, Buffer[0]);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
	return AL_FALSE;

//set sound Loop ON
alSourcei(Sources[0], AL_LOOPING, AL_TRUE);//Loop ON
if ( Error != AL_NO_ERROR )
	return AL_FALSE;

//play a  source
alSourcePlay (Sources[0]);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
	return AL_FALSE;

return AL_TRUE;
}
```

```
ALvoid ExitPlay(){

	alSourcef(Sources[0],AL_GAIN,0.1f);
	alutSleep(1);
	alSourceStop(Sources[0]);
	alDeleteSources(1,Sources);
	alDeleteBuffers(1,Buffer);
}
```

/*SystemWindow.h*/

```
#ifndef SYSTEMWINDOW_H_
#define SYSTEMWINDOW_H_

//Ran widows - Load Data -All Setup For OpenGL-Initialize AL
//Initialization
BOOL WindowsRAN(char* title, int width, int height, BYTE bits);
```

```
/
/Create OpenGL Window
BOOL CreateGLWindow(char* title, int width, int height, BYTE bits);
//Exit - Close OpenGLWindow
GLvoid ExitGLWindow();

//Resize The OpenGL Scene Whenever The Window Has Been Resized.
GLboolean ReSizeGLScene(GLsizei width, GLsizei height);
//Initialization OpenGL
GLboolean InitOpenGL();

#endif /*SYSTEMWINDOW_H_*/
```

/*SystemWindow.cpp*/

```
#include "Include.h"

HGLRC           hRC=NULL;     // Open GL Rendering Context
HDC             hDC=NULL;     // Open GL GDI(Graphics Device
                              //Interface)Device Context
HWND            hWnd=NULL;    // Holds Window Handle
HINSTANCE       hInstance;    // Holds The Instance Of The
Application

BOOL WindowsRAN(char* title, int width, int height, BYTE bits)
{
//Ran widows-Load Data-All Setup For OpenGL- AL Initialize
//Setup AL Sound Full
if (!ALSound ()){
MessageBoxA (NULL, "Δεν βρέθηκε κανένας ελεγκτής ήχου στο σύστημά
σας.\n\n\n" "Βεβαιωθείτε ότι η συσκευή σας δουλεύει κανονικά και
εγκαταστήστε το πιο\n""πρόσφατο οδηγό για τη κάρτα ήχου σας.\n",
NULL,MB_ICONWARNING | MB_TASKMODAL );
      return FALSE;
}
//Sound Loading
if(!LoadWavSound ()){//Load a .WAV sound
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
MessageBoxA (NULL,"Δεν είναι δυνατή η προσπέλαση των αρχείων
ήχου.\n\n\n" "Η επανεγκατάσταση της εφαρμογής ίσως λύσει το πρόβλημα"
, NULL,MB_ICONWARNING | MB_TASKMODAL );

      return FALSE;
}
//Setup And Create GL Window
if (!CreateGLWindow(title,width,height,bits)){
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitGLWindow();

MessageBoxA (NULL,"Δεν ειναι δυνατή η δημιουργία ενός window.\n\n\n"
"Βεβαιωθείτε ότι η συσκευή σας δουλεύει κανονικά και εγκαταστήστε τον
πιο\n""πρόσφατο οδηγό για την κάρτα γραφικών σας.\n"
, NULL,MB_ICONWARNING | MB_TASKMODAL );

     return FALSE;
}
```

```
//Texture Loading
if (!LoadGLTextures())
{
	CloseALUT();//Exit-Close ALUT
	CloseOpenAL();//Exit-Close OpenAL
	ExitGLWindow();

MessageBoxA (NULL, "Δεν είναι δυνατή η προσπέλαση των αρχείων
εικόνας.\n\n\n""Η επανεγκατάσταση της εφαρμογής ίσως λύσει το
πρόβλημα.\n", NULL,MB_ICONWARNING | MB_TASKMODAL );
		return FALSE;
}
//All Property Setup For OpenGL
if (!InitOpenGL()){
	//!!!ERROR:Setup For OpenGL!!!
	CloseALUT();//Exit-Close ALUT
	CloseOpenAL();//Exit-Close OpenAL
	ExitGLWindow();

MessageBoxA (NULL, "Δεν είναι δυνατή η δημιουργία ενός GL
window.\n\n\n"Βεβαιωθείτε ότι η συσκευή σας δουλεύει κανονικά και
εγκαταστήστε τον πιο\n""πρόσφατο οδηγό για την κάρτα γραφικών σας.\n"
, NULL,MB_ICONWARNING | MB_TASKMODAL );
	return FALSE;
}

if(!SoundPlay()){
	ExitPlay();
	CloseALUT();//Exit-Close ALUT
	CloseOpenAL();//Exit-Close OpenAL
	ExitGLWindow();

MessageBoxA (NULL, "Δεν βρέθηκε κανένας έγκυρος ελεγκτής ήχου στο
σύστημά σας που να υποστηρίζει την εφαρμογή.\n\n\n""Βεβαιωθείτε ότι η
συσκευή σας δουλεύει κανονικά και εγκαταστήστε τον πιο \n""πρόσφατο
οδηγό για τη κάρτα ήχου σας.\n",NULL,MB_ICONWARNING | MB_TASKMODAL );
	return FALSE;
}

return TRUE;
}
```

```
BOOL CreateGLWindow(char* title, int width, int height, BYTE bits){
//This Code Creates  OpenGL Window.  Parameters Are:
//title     - Title To Appear At The Top Of The Window
//width     - Width Of The GL Window Or Fullscreen Mode
//height    - Height Of The GL Window Or Fullscreen Mode
//bits      - Number Of Bits To Use For Color (8/16/24/32)

GLuint PixelFormat;// Holds The Results After Searching For A Match

WNDCLASS     wc;        // Windows Class Structure
DWORD       dwExStyle;  // Window Extended Style
DWORD       dwStyle;    // Window Style

// Grabs Rectangle Upper Left / Lower Right Values
RECT        WindowRect;

WindowRect.left=(long)0;       // Set Left Value To 0
```

```
WindowRect.right=(long)width;  // Set Right Value To Requested Width
WindowRect.top=(long)0;        // Set Top Value To 0
WindowRect.bottom=(long)height;//Set Bottom Value To Requested Height

hInstance    = GetModuleHandle(NULL);//Grab An Instance For Our Window

//CS_HREDRAW :Redraws the entire window if a movement or size
//adjustment changes the width of the client area.
//CS_VREDRAW :Redraws the entire window if a movement or size
//adjustment changes the height of the client area.
//CS_OWNDC:Allocates a unique device context for each window in the
//class
//Info:http://msdn2.microsoft.com/en-us/library/ms633574.aspx
wc.style          = CS_HREDRAW | CS_VREDRAW | CS_OWNDC;
wc.lpfnWndProc    = (WNDPROC) WndProc;// WndProc Handles Messages
wc.cbClsExtra     = 0;         // No Extra Window Data
wc.cbWndExtra     = 0;         // No Extra Window Data
wc.hInstance      = hInstance;// Set The Instance
wc.hIcon          = LoadIcon(NULL, IDI_WINLOGO);// Load Default Icon
wc.hCursor        = LoadCursor(NULL, IDC_ARROW);// Load Arrow Pointer
wc.hbrBackground  = NULL;      // No Background Required For GL
wc.lpszMenuName   = NULL;      // We Don't Want A Menu
wc.lpszClassName  = "OpenGL"; // Set The Class Name


if (!RegisterClass(&wc)){//Attempt To Register The Window Class
      ExitGLWindow();
      return FALSE;//!!!ERROR:Failed To Register The Window Class
}

//APPWINDOW:Forces a top-level window onto the taskbar when the
//window is visible.
//WINDOWEDGE:Specifies that a window has a border with a raised edge.
dwExStyle=WS_EX_APPWINDOW | WS_EX_WINDOWEDGE;


//Creates an overlapped window with the WS_OVERLAPPED, WS_CAPTION,
//WS_SYSMENU, WS_THICKFRAME, WS_MINIMIZEBOX, and WS_MAXIMIZEBOX
//styles.
dwStyle=WS_OVERLAPPEDWINDOW;// Windows Style

//AdjustWindowRectEx:Calculates The Required Size Of The Window
//Rectangle, Based On The Desired Size Of The Client Rectangle. The
//Window Rectangle Can Then Be Passed To The CreateWindowEx Function
//To Create a Window Whose Client Area Is The Desired Size.
//Info:http://msdn2.microsoft.com/en-us/library/ms632667.aspx


if (!AdjustWindowRectEx(&WindowRect, dwStyle, FALSE, dwExStyle)){
      ExitGLWindow();
      return FALSE;//!!!ERROR:Adjust Window To True Requested Size
}


// Create The Window
hWnd=CreateWindowEx(dwExStyle, //Extended Style For The Window
                    "OpenGL",  //Class Name
                     title,    //Window Title
                     dwStyle | //Defined Window Style
```

```
                        WS_CLIPSIBLINGS |//Required Window Style
                        WS_CLIPCHILDREN,  //Required Window Style
                        0, 0,             //Window Position
                       //Calculate Window Width
                       WindowRect.right-WindowRect.left,
                       //Calculate Window Height
                       WindowRect.bottom-WindowRect.top,
                       NULL,           //No Parent Window
                       NULL,           //No Menu
                       hInstance,      //Instance
                       NULL);          //Dont Pass Anything To WM_CREATE


if (!hWnd){//!!!ERROR:Window Creation Error.!!!
            ExitGLWindow();
            return FALSE;
}


//PIXELFORMATDESCRIPTOR:Describes The Pixel Format Of A Drawing
//Surface.
//Info:http://msdn2.microsoft.com/en-us/library/ms537569.aspx
//Choose a Format That Supports OpenGL and Double
//Buffering,RGBA(red,green,blue,alpha channel).
//Find a Pixel Format That Matches The Bits we Decided on (16bit,
//24bit, 32bit). Set Up a 16bit Z-Buffer.
//The Remaining Parameters Are Either Not Used Or Are Not Important
//(Aside From The Stencil Buffer And The (Slow) Accumulation Buffer).
//pfd Tells Windows How We Want Things To Be
static      PIXELFORMATDESCRIPTOR pfd=
{
     //Size Of This Pixel Format Descriptor
      sizeof(PIXELFORMATDESCRIPTOR),
      1,                              //Version Number
      PFD_DRAW_TO_WINDOW |            //Format Must Support Window
      PFD_SUPPORT_OPENGL |            //Format Must Support OpenGL
      PFD_DOUBLEBUFFER,               //Must Support Double Buffering
      PFD_TYPE_RGBA,                  //Request An RGBA Format
      bits,                           //Select Our Color Depth
      0, 0, 0, 0, 0, 0,               //Color Bits Ignored
      0,                              //No Alpha Buffer
      0,                              //Shift Bit Ignored
      0,                              //No Accumulation Buffer
      0, 0, 0, 0,                     //Accumulation Bits Ignored
      bits,                           //16Bit Z-Buffer (Depth Buffer)
      0,                              //No Stencil Buffer
      0,                              //No Auxiliary Buffer
      PFD_MAIN_PLANE,                 //Main Drawing Layer
      0,                              //Reserved
      0, 0, 0                         //Layer Masks Ignored
      };

hDC=GetDC(hWnd);//Get A GL Device Context
if (!hDC){
      ExitGLWindow();
      return FALSE;//!!!ERROR:Can't Create A GL Device Context!!!
}
```

```
//ChoosePixelFormat: Attempts To Match An Appropriate Pixel Format
//Supported By a Device Context To a Given Pixel Format
//Specification.
//Info:http://msdn2.microsoft.com/en-us/library/ms537556.aspx
PixelFormat=ChoosePixelFormat(hDC,&pfd);
if (!PixelFormat){
      ExitGLWindow();
      return FALSE;//!!!ERROR:Can't Find A Matching PixelFormat!!!
}

if(!SetPixelFormat(hDC,PixelFormat,&pfd)){//Set The Pixel Format
      ExitGLWindow();
      return FALSE;//!!!ERROR:Can't Set The PixelFormat!!!
}

hRC=wglCreateContext(hDC);//Create A Rendering Context
if (!hRC){
      ExitGLWindow();
      return FALSE;//!!!ERROR:Can't Create A GL Rendering Context!!!
}

if(!wglMakeCurrent(hDC,hRC)){//set active The Rendering Context
      ExitGLWindow();
      return FALSE;//!!!ERROR:Can't Activate The GL Rendering
                   //Context!!!
}

ShowWindow(hWnd,SW_SHOW);//Show The Window
SetForegroundWindow(hWnd);//Slightly Higher Priority
SetFocus(hWnd);//Sets Keyboard Focus To The Window
ReSizeGLScene(width, height);//Set Up Our Perspective GL Screen

return TRUE;
}
```

```
GLvoid ExitGLWindow()
{//Exit - Close Window

HGLRC hRC= wglGetCurrentContext();//Get current GL Rendering Context
HDC   hDC= wglGetCurrentDC();       //Get current GL Device Context

if (hRC){//Rendering Context
      wglMakeCurrent(NULL,NULL);//makes NULL The DC And RC Contexts
      wglDeleteContext(hRC);//Delete Rendering Context
      hRC=NULL;// Set Rendering Context To NULL
}

if (hDC && !ReleaseDC(hWnd,hDC)){//Release The Device Context
      hDC=NULL;//Set Device Context  NULL
}

if (hWnd && !DestroyWindow(hWnd)){//Destroy The Window
      hWnd=NULL;//Set Window Handle To NULL
}

if (!UnregisterClass("OpenGL",hInstance)){//Unregister Class
      hInstance=NULL;//Set Unregister Class To NULL
}
}
```

```
GLboolean ReSizeGLScene(GLsizei width, GLsizei height)
{

//Resize The OpenGL Scene Whenever The Window Has Been Resized.
//Example:ReSizeGLScene(800,600)

GLenum                  ErrorGL;//Save GL Error
GLboolean         bReturn = GL_TRUE;

//Clear error code
//http://msdn2.microsoft.com/en-us/library/ms537109(VS.85).aspx
glGetError();

if (height==0)// Prevent A Divide By Zero By
{
      height=1;// Making Height Equal One
}

glViewport(0,0,width,height);// Reset The Current Viewport
ErrorGL=glGetError();//!!!Error:Failure Reset The Current Viewport!!!
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

glMatrixMode(GL_PROJECTION);// Select The Projection Matrix
//GL_PROJECTION:Applies subsequent matrix operations to the
//projection matrix stack.
ErrorGL=glGetError();//!!!Error:Failure Initialize The Matrix State
if (ErrorGL  != GL_NO_ERROR)
    bReturn = GL_FALSE;

//Replace The Current Matrix With The Identity Matrix
glLoadIdentity();
ErrorGL=glGetError();//!!!Error:Failure Replace Matrix!!!
if (ErrorGL  != GL_NO_ERROR)
    bReturn = GL_FALSE;

// Calculate The Aspect Ratio Of The Window
gluPerspective(45.0f,(GLfloat)width/(GLfloat)height,0.1f,100.0f);
//set up a perspective  projection matrix
ErrorGL=glGetError();//!!!Error:Failure ......!!!
if (ErrorGL  != GL_NO_ERROR)
     bReturn = GL_FALSE;

glMatrixMode(GL_MODELVIEW);//Select The Modelview Matrix
//GL_MODELVIEW: Applies subsequent matrix operations to
//the modelview matrix stack.
ErrorGL=glGetError();//!!!Error:Failure Initialize The Matrix State
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

glLoadIdentity();//Replace The Current Matrix With The IdentityMatrix
ErrorGL=glGetError();//!!!Error:Failure Replace Matrix!!!
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

return bReturn;

}
```

```
/Red,Blue,Green,Alpha Value

//Ambient Light Values - Without an ambient light, spots where there
//is no diffuse light will appear very dark.
GLfloat LightAmbient[]=  { 0.5f, 0.5f, 0.5f, 1.0f };

//Diffuse Light Values- A diffuse light this bright lights up the
//front of the crate nicely.
GLfloat LightDiffuse[]=  { 1.0f, 1.0f, 1.0f, 1.0f };

//x,y,z,the last number at 1.0f.tells OpenGL the designated
coordinates are the position of the light source.
//Light Position-position the light off the screen,towards the
//viewe(2.0f),
GLfloat LightPosition[]= { 0.0f, 0.0f, 2.0f, 1.0f };
GLboolean InitOpenGL()
{//Initialization  OpenGL

GLenum                  ErrorGL= GL_NO_ERROR;//Save GL Error
GLboolean         bReturn = GL_TRUE;

glGetError();//Clear error code

//Enable Texture Mapping
glEnable(GL_TEXTURE_2D);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

//Enable Smooth Shading
glShadeModel(GL_SMOOTH);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

//Black Background(Red,Green,Blue,Alpha)The Initial Values Are All 0.
glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

// Depth Buffer Setup. The Initial Value is 1.
glClearDepth(1.0f);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

//Enables Depth Testing
glEnable(GL_DEPTH_TEST);
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

//Specifies The Value Used For Depth-Buffer Comparisons.
//Specifies The Function Used To Compare Each Incoming Pixel z
//Value With The z Value Present In The Depth Buffer. The Comparison
//Is Performed Only If Depth Testing is Enabled.
//GL_LEGUAL:Passes if the incoming z value is less than or equal to
//the stored z value.
```

```
//Info:http://msdn2.microsoft.com/en-us/library/ms537051.aspx
glDepthFunc(GL_LEQUAL);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

//Specify Implementation-Specific Hints
//GL_PERSPECTIVE_CORRECTION_HINT:Indicates the quality of color and
//texture coordinate interpolation.
//GL_NICEST:The most correct,or highest quality,option should be
//chosen.
glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;
//set up the lighting
//Setup The Ambient Light.Set the amount of ambient
//light that light1 will give off.
//Half intensity ambient light
glLightfv(GL_LIGHT1, GL_AMBIENT, LightAmbient);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

//Setup The Diffuse Light.Set up the amount of diffuse
//light that light number one will give off
// Full intensity white light
glLightfv(GL_LIGHT1, GL_DIFFUSE, LightDiffuse);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

// Position The Light.Set the position of the light.2
//unit towards the viewer
glLightfv(GL_LIGHT1, GL_POSITION,LightPosition);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

// Enable Light One
glEnable(GL_LIGHT1);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

return bReturn;
}
```

/*main.cpp*/

```
#include "Include.h"

bool  done=FALSE;       // Bool Variable To Exit Loop - Windows
bool  keys[256];        // Array Used For The Keyboard Routine
MSG   msg;              // Windows Message Structure
bool  active=TRUE;      // Window Active Flag Set To TRUE By Default
bool  fp;                     // F Pressed?
bool  cp;                     // C Pressed?
```

```
GLuint      filter = 2;         // Which Filter To Use -High

GLfloat PosUD  = 0.135f;        //Position Up -Down
GLfloat PosIO  = 0.135f;        //Position Inside - Outsude
GLfloat PosLR;                  //Position Right -Left

GLfloat ypos;                   //Position Up -Down
GLfloat zpos;                   //Position Inside - Outside
GLfloat xpos;                   //Position Right -Left
```

```
//The WindowProc function is an application-defined function that
//processes messages sent to a window. The WNDPROC type defines a
//pointer to this callback function. WindowProc is a placeholder for
//the application-defined function name.
//Info:http://msdn2.microsoft.com/en-us/library/ms633573.aspx

LRESULT CALLBACK WndProc(HWND  hWnd,      //Handle For This Window
                         UINT  uMsg,      //Message For This Window
                         WPARAM wParam,   //Additional Message Info
                         LPARAM lParam)   //Additional Message Info
{



      switch (uMsg)//Check For Windows Messages
      {
      case WM_ACTIVATE://Watch For Window Activate Message
      {
            if (!HIWORD(wParam)){//Check Minimization State
                  active=TRUE;//Program Is Active
            }
            else{
                  active=FALSE;//Program Is No Longer Active
            }
            return 0;
      }
      //WM_SYSCOMMAND:A window receives this message when the
      //user chooses a command from the Window menu (formerly known
      //as the system or control  menu) or when the user chooses the
      //maximize button, minimize button, restore button, or close
       //button.
      //Info:http://msdn2.microsoft.com/en-us/library/ms646360.aspx
      case WM_SYSCOMMAND://Intercept System Commands
      {
            switch (wParam){//Check System Calls
                  //Screensaver Trying To Start
                  case SC_SCREENSAVE:
                  //Monitor Trying To Enter Powersave.
                  case SC_MONITORPOWER:
                 // By Returning 0 We Prevent Those Things From
                 // Happening.
                  return 0;
            }
      break;
      }

//Info:http://msdn2.microsoft.com/en-us/library/ms674887.aspx
//Sent As A Signal That A Window Or An Application Should Terminate.
      case WM_CLOSE:
```

```
	{//Send A Quit Message.The Variable Done Will Be Set To TRUE,
	//The Main Loop In WinMain() Will Stop, And The Program Will
	//Close.
		PostQuitMessage(0);
		return 0;
	}

//The WM_KEYDOWN message is posted to the window with the keyboard
//focus when a nonsystem key is pressed. A nonsystem key is a key
//that is pressed when the ALT key is not pressed.
//Info:http://msdn2.microsoft.com/en-us/library/ms646280.aspx
	case WM_KEYDOWN:
	{
//If a key is being held down we can find out what key it is by
//reading wParam.We then make that keys cell in the array keys[ ]
//become TRUE.That way I can read the array later on and find out
//which keys are being held down.This allows more than one key to be
//pressed at the same time.
		keys[wParam] = TRUE;// If So, Mark It As TRUE
		return 0;
	}
//Info:http://msdn2.microsoft.com/en-us/library/ms646281.aspx
	case WM_KEYUP:
	{
//If a key has been released we find out which key it was by reading
//wParam. We then make that keys cell in the array keys[] equal
//FALSE.That way when I read the cell for that key I'll know if it's
//still being held down or if it's been released.
//Each key on the keyboard can be represented by a number from 0-255.
//When I press the key that represents the number 40 for example,
//keys[40] will become TRUE. When I let go, it will become FALSE.
		keys[wParam] = FALSE;
		return 0;
	}
	case WM_SIZE://Resize The OpenGL Window
	{

	//LoWord=Width, HiWord=Height
	ReSizeGLScene(LOWORD(lParam),HIWORD(lParam));
		return 0;
	}
	}//END switch (uMsg)

//Pass All Unhandled Messages To DefWindowProc
//Calls the default window procedure to provide default processing
//for any window messages that an application does not process. This
//function ensures that every message is processed.
//Info:http://msdn2.microsoft.com/en-us/library/ms633572.aspx
return DefWindowProc(hWnd,uMsg,wParam,lParam);
}
```

```
int WINAPI WinMain(HINSTANCE  hInstance,    // Instance
                   HINSTANCE  hPrevInstance,// Previous Instance
                   LPSTR      lpCmdLine,   // Command Line Parameters
                   int        nCmdShow)    // Window Show State
{

if (!WindowsRAN("Moving Source",640,480,32)){
		return 0;//Quit If Window Was Not Created
	}
```

```
while(!done)
{

      //Is There A Message Waiting?
      if (PeekMessage(&msg,NULL,0,0,PM_REMOVE)){
      //Have We Received A Quit Message?
            if (msg.message==WM_QUIT){
                  done=TRUE;        //If So done=TRUE
            }
            else{ //If Not, Deal With Window Messages
                  TranslateMessage(&msg); //Translate The Message
                  DispatchMessage(&msg);  //Dispatch The Message
            }
      }
      else{//If There Are No Messages
      // Draw The Scene.  Watch For ESC Key.
            if (active && keys[VK_ESCAPE]){// Quit Received?
                        done=TRUE;  // ESC
             }
            else{//Star DrawGLScene

                // Not Time To Quit, Update Screen
                  DrawGLScene();// Draw The Scene
                 //Get current GL Device Context
                  HDC   hDC= wglGetCurrentDC();

//Specifies a device context. If the current pixel format for the
//window referenced by thisdevice context includes a back buffer,
//the function exchanges the front and back buffers.
//http://msdn2.microsoft.com/en-us/library/ms537551(VS.85).aspx
                  SwapBuffers(hDC); //Swap Buffers (Double Buffering)

//********************** Filtre Toggle ***************************//
                  if (keys['F'] && !fp){  // Is F Key Being Pressed?
                        fp=TRUE;          // fp Becomes TRUE
                        filter+=1;  // filter Value Increases By One
                        if (filter>2){// Is Value Greater Than 2?
                        filter=0;   / If So, Set filter To 0
                        }
                  }
                  if (!keys['F']){  // Has F Key Been Released?
                        fp=FALSE;   // If So, fp Becomes FALSE
                  }
//************************************************************//
                  if (keys[VK_PRIOR]){// Is Page Up Being Pressed
                        PosUD+=0.003f;// If So, Move UP
                        if(PosUD>0.771001f){ //if Max up
                              PosUD = 0.771001f;
                        }
                        else{
                              ypos+=0.03f;
                        }
                  }
                  if (keys[VK_NEXT]){// Is Page Down Being Pressed
                        PosUD-=0.003f;// If So, Move Down
                        if(PosUD<-0.501000f ){ //if Max Down
                              PosUD = -0.501000f;
                        }
                        else{
```

```
                                ypos-=0.03f;
                        }
                }
                if (keys[VK_UP]){ // Is Up Arrow Being Pressed
                        PosIO+=0.003f; // If So, Move Into The Screen
                        if(PosIO>0.771001f){//if Max up
                                PosIO = 0.771001f;
                        }
                        else{
                                zpos-=0.03f;
                        }
                }
                if (keys[VK_DOWN]){// Is Down Arrow Being Pressed
                        PosIO-=0.003f;//If So, Move
                        if(PosIO<-0.501000f){  //if Max Down
                                PosIO = -0.501000f;
                        }
                        else{
                                zpos+=0.03f;
                        }
                }
                if (keys[VK_RIGHT]){// Is Right Arrow Being Pressed
                        PosLR+= 0.003f;// If So, Move Right
                        if(PosLR>0.771001f){ //if Max Right
                                PosLR = 0.771001f;
                        }
                        else {
                                xpos+=  0.03f;
                        }
                }
                if (keys[VK_LEFT]){// Is Left Arrow Being Pressed
                        PosLR-= 0.003f;// If So, Move Left
                        if(PosLR<-0.771001f){ //if Max Left
                                PosLR = -0.771001f;
                        }
                        else {
                                xpos-=  0.03f;
                        }
                }
                if (keys['C']&& !cp){
                        cp =TRUE;
                        FILE *file;//File Name
                        char *Getlog = new char [500];
                       //file = fopen ("Sound.log","a");
                        fopen_s(&file,"Sound.log","a");
                        if (file){
                        sprintf_s (Getlog,500,"xpos: %f  ypos :
                                  %f zpos : %f\n",xpos,ypos,zpos);
                        fputs(Getlog,file);
                        fclose (file);
                        }
                }
                if (!keys['C']){
                        cp=FALSE;
                }

            }//End  drawGLScene
        }//End If There Are No Messages
}//End while(!done)
```

```
/* ============================================================== */
/*                 Exit -Close System                             */
/* ============================================================== */
      ExitPlay();
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitGLWindow();//Close Window -OpenAL
return (msg.wParam);//Exit The Program.Return PostQuitMessage(0);
}
```

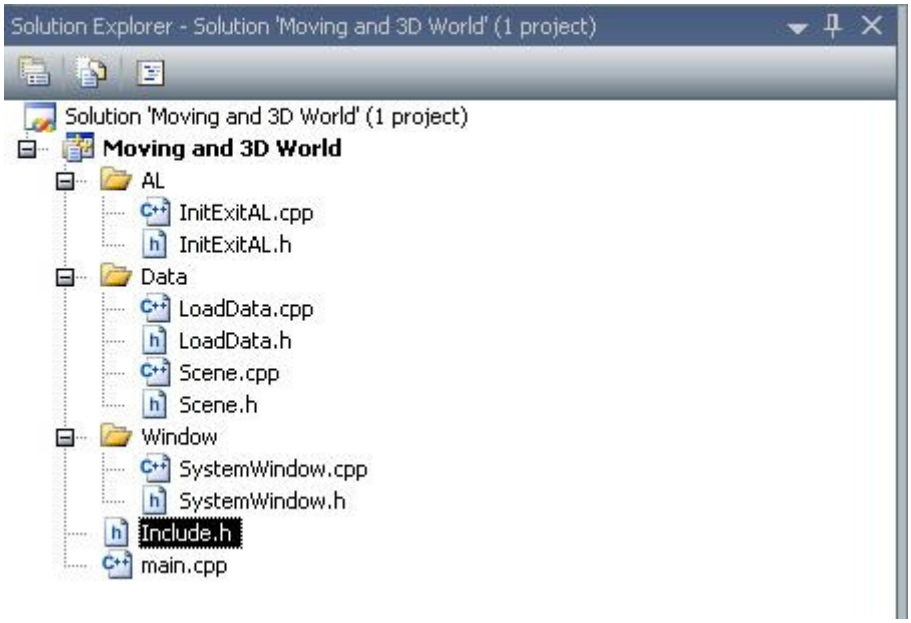
9. Example: Moving and 3d world (Soundspace)

Το παράδειγμα εννιά αποτελεί μια τρισδιάστατη μοντελοποίηση του ηχητικού χώρου (Soundspace). Για την δημιουργία του χρησιμοποιήθηκε η OpenAL, το win32 API και η OpenGL. Είναι γραμμένο σε γλώσσα C++ με τον Microsoft Visual C++ 2008 Express Edition. Ο λόγος για την διαφοροποίηση αυτή έχει άμεση σχέση με το win32 API, μιας και δεν υποστηρίζεται πλήρως από το GCC. Για την δημιουργία του παραδείγματος χρησιμοποιήθηκαν πληροφορίες από τον δικτυακό τόπο του the Forger's Win32 API Tutorial (http://www.winprog.org/tutorial/), ενώ πληροφορίες, καθώς και τμήματα κώδικα, όσον αφορά την δομή του window και της OpenGL, προήλθαν από τον δικτυακό τόπο Nehe Productions (http://nehe.gamedev.net/) Lessons 01 -10.

Το παράδειγμα αυτό δημιουργεί ένα window σε διαστάσεις 1024x768 με ανάλυση 32bit χρωμάτων. Στην συνέχεια ενφανίζει τέσσερεις κύβους με διαφορετικά textures, που έχουν τοποθετημένο στο κέντρο τους από έναν ήχο. Ο χρήστης μπορεί να μετακινηθεί στον χώρο χρησιμοποιώντας τα βελάκια και τα Page up-down του πληκτρολογίου. Καθώς ο χρήστης μετακινείται μέσα στον χώρο, αλλάζει η ηχητική του θέση και παρουσιάζεται το ηχητικό περιβάλλον, σύμφωνα με την θέση που βρίσκεται την δεδομένη στιγμή. Όταν ο χρήστης μπει μέσα σε έναν κύβο οι άλλοι ήχοι αποκόπτονται, ενώ ο συγκεκριμένος ήχος του κύβου που εισήλθε ακούγεται με reverb. Ουσιαστικά γίνεται προσομοίωση του ηχητικού χώρου που εισήλθε ο χρήστης. Αντίστοιχα όταν ο χρήστης βγει από έναν κύβο, το ηχητικό περιβάλλον επανέρχεται στην προηγούμενη κατάστασή του.

Το συγκεκριμένο παράδειγμα διαθέτει κάποιους βασικούς περιορισμούς. Δεν θα εκτελεστεί σε ένα σύστημα που δεν διαθέτει OpenGL 2.0 και EAX 2.0. Και στις δύο περιπτώσεις γίνεται έλεγχος από το πρόγραμμα και εξάγει μηνύματα σφάλματος, που ενημερώνουν τον χρήστη ότι πρέπει να αναβαθμίσει την κάρτα ήχου ή γραφικών του. Όπως ήδη έχουμε προαναφέρει το EAX 2.0 αποτελεί πλέον στάνταρ για τις κάρτες ήχου, ενώ οι επόμενες εκδόσεις δεν είναι δεδομένο ότι υποστηρίζονται. Για αυτόν τον λόγο χρησιμοποιήθηκαν μόνο οι δυνατότητες του EAX 2.0 και επομένως το παράδειγμα αυτό δεν αποτελεί ένα παράδειγμα πολλαπλών περιβαλλόντων. Όμως ακόμα και για το EAX 5.0 η δομή του κώδικα θα ήταν ίδια, απλά δεν θα τοποθετούνταν τα ίδια auxiliary effect slot σε όλους τους κύβους.

Στην παρακάτω εικόνα παρουσιάζεται η δομή του κώδικα :



Όπως φαίνεται το πρόγραμμα χωρίζεται σε τρεις υποφακέλους. Σε αυτό το παράδειγμα θα αναλυθούν μόνο τα τμήματα του κώδικα που είναι καινούργια και δεν αναλύονται στο παράδειγμα οκτώ.

Περιληπτικά αναφέρουμε ότι στον υποφάκελο AL υπάρχουν οι συναρτήσεις για το initialize και exit των AL και ALUT. Στον υποφάκελο Data το αρχείο loadData περιέχει όλες τις συναρτήσεις που απαιτούνται για να γίνουν load τα textures και οι ήχοι, ενώ το αρχείο Scene είναι ο πυρήνας όλου του προγράμματος, δηλαδή είναι το μέρος που δημιουργούνται τα γραφικά και η geodata για το ηχητικό περιβάλλον. Ουσιαστικά περιέχει όλες τις λειτουργίες που πραγματοποιούνται, αφού δημιουργηθεί το παραθυρικό περιβάλον. Στον υποφάκελο window υπάρχουν όλες οι συναρτήσεις που χρειάζονται για την δημιουργία ενός window GL, καθώς και μια συνάρτηση ρύθμισης για το μέγεθος του παράθυρου - ReSize windows.

Το πρόγραμμα ουσιαστικά ξεκινάει και πάλι, εκτελώντας την παρακάτω γραμμή εντολών

```
if (!WindowsRAN("Moving and 3D World",1024,768,32)){
            return 0;//Quit If Window Was Not Created
      }
```

Η συνάρτηση αυτή εκτελεί τις λειτουργίες που έχουμε προαναφέρει στο παράδειγμα οκτώ και οι διαφορές επικεντρώνονται στα εξής σημεία:

- Η συνάρτηση LoadWavSound () φορτώνει τέσσερα αρχεία ήχου σε τέσσερις buffers .

- Η συνάρτηση LoadGLTextures() φορτώνει τέσσερα αρχεία εικόνας.

- Η συνάρτηση SoundPlay() αρχικά κάνει Setup των EFX Functions. Στην συνέχεια δημιουργεί τέσσερα sources και τοποθετεί τους buffers στους sources. Έπειτα θέτει τις θέσεις που πρέπει να βρίσκονται οι sources, τους θέτει σε Loop On, και εντέλει, ξεκινάει την δημιουργία EAX object. Αν οποιαδήποτε από αυτές τις λειτουργίες επιστρέψει ένα σφάλμα, όλη η διαδικασία σταματάει και το πρόγραμμα κλείνει. Σε αυτήν την περίπτωση το πιθανότερο είναι μην υποστηρίζεται το EAX 2.0 από την κάρτα ήχου.

Στην συνάρτηση SoundPlay() τοποθετήθηκαν και κάποιες «δοκιμές» για το EAX. Όλες οι λειτουργίες που θα εκτελεστούν αργότερα σε πραγματικό χρόνο ελέγχονται για σφάλματα στο συγκεκριμένο τμήμα του κώδικα. Στην περίπτωση που δεν υποστηρίζεται κάποια από αυτές, η συνάρτηση και πάλι επιστρέφει FALSE και το πρόγραμμα δεν εκτελείται. Αν όμως δεν παρουσιαστεί κανένα πρόβλημα, ξεκινά την εκτέλεση των ήχων. Ο κώδικας των συγκεκριμένων «δοκιμών» φαίνεται παρακάτω:

```
//Test Code 1 : Set Source into Auxilary
alSource3i(Sources[0], AL_AUXILIARY_SEND_FILTER, EffectSlot, 0, Filter);
alSource3i(Sources[1], AL_AUXILIARY_SEND_FILTER, EffectSlot, 0, Filter);
alSource3i(Sources[2], AL_AUXILIARY_SEND_FILTER, EffectSlot, 0, Filter);
alSource3i(Sources[3], AL_AUXILIARY_SEND_FILTER, EffectSlot, 0, Filter);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//Test Code 2: if Test1 is ok -Remove Source and test if Remove is ok ?
alSource3i(Sources[0],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
AL_FILTER_NULL);
alSource3i(Sources[1],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
AL_FILTER_NULL);
alSource3i(Sources[2],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
AL_FILTER_NULL);
alSource3i(Sources[3],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
AL_FILTER_NULL);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;



//Test Code 3: set Filter into Source
alSourcei(Sources[0], AL_DIRECT_FILTER, Filter);
alSourcei(Sources[1], AL_DIRECT_FILTER, Filter);
alSourcei(Sources[2], AL_DIRECT_FILTER, Filter);
alSourcei(Sources[3], AL_DIRECT_FILTER, Filter);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;
```

```
///Test Code 4: if Test3 is ok -Remove Filter and test if Remove is ok ?
alSourcei(Sources[0], AL_DIRECT_FILTER, AL_FILTER_NULL);
alSourcei(Sources[1], AL_DIRECT_FILTER, AL_FILTER_NULL);
alSourcei(Sources[2], AL_DIRECT_FILTER, AL_FILTER_NULL);
alSourcei(Sources[3], AL_DIRECT_FILTER, AL_FILTER_NULL);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;
```

Επιστρέφοντας στην συνάρτηση main, ξεκινάμε την ρουτίνα που θα τερματιστεί μόνο αν η bool done γίνει TRUE, όπως έχουμε προαναφέρει.

```
while(!done)
{

      //Is There A Message Waiting?
      if (PeekMessage(&msg,NULL,0,0,PM_REMOVE)){
      //Have We Received A Quit Message?
            if (msg.message==WM_QUIT){
                  done=TRUE;  //If So done=TRUE
            }
            else{ //If Not, Deal With Window Messages
                  TranslateMessage(&msg); //Translate The Message
                  DispatchMessage(&msg);  //Dispatch The Message
            }
      }
      else{//If There Are No Messages
          if (active && keys[VK_ESCAPE]){
               done=TRUE;//Quit
          }
         else{//Star DrawGLScene
              // Not Time To Quit, Update Screen
              DrawGLScene();// Draw The Scene
```

Στην συνέχεια, εκτελούμε την συνάρτηση DrawGLScen, που περιέχει οτιδήποτε αλλάζει μέσα στο παράθυρο, όσον αφορά τους ήχους και το γραφικό περιβάλλον. Η συνάρτηση αυτή θα αναλυθεί αφού ολοκληρωθεί η ανάλυση της ρουτίνας. Επομένως όταν εκτελεστεί η συνάρτηση, ελέγχουμε το πάτημα των πλήκτρων Page up–down, καθώς και τα βελάκια του πληκτρολογίου. Στην περίπτωση που η LRESULT CALLBACK wndProc μας έχει δώσει το μήνυμα ότι κάποιο από τα συγκεκριμένα πλήκτρα έχουν πατηθεί, τότε αυξάνουμε ή μειώνουμε την αντίστοιχη τιμή.

```
      if (keys[VK_PRIOR]){// Is Page Up Being Pressed
            ypos+=0.03f;  // If So, Move UP
      }
      if (keys[VK_NEXT]){// Is Page Down Being Pressed
            ypos-=0.03f; // If So, Move Down
      }
```

```
        if (keys[VK_UP]){ // Is Up Arrow Being Pressed
              zpos-=0.03f;// If So, Move Into The Screen
        }
        if (keys[VK_DOWN]){// Is Down Arrow Being Pressed
              zpos+=0.03f;// If So, Move Towards The Viewer
        }
        if (keys[VK_RIGHT]){// Is Right Arrow Being Pressed
              xpos+= 0.1f; // If So, Move Right
        }
        if (keys[VK_LEFT]){// Is Left Arrow Being Pressed
              xpos-= 0.1f;// If So, Move Left
        }
}//End  drawGLScene
}//End If There Are No Messages
}//End while(!done)
```

Οι τιμές αυτές θα προσδιορίζουν την θέση του χρήστη στον εικονικό κόσμο και έχουν δηλωθεί στην αρχή του προγράμματος, όπως φαίνεται στο παρακάτω τμήμα του κώδικα:

```
GLfloat zpos;                          //Position Int - Out
GLfloat ypos;                          //Position Up -Down
GLfloat xpos;                          //Position Right -Left
```

Εισάγονται στο αρχείο Scene.cpp και ερμηνεύονται ως θέση του χρήστη:

```
//extern form Main.cpp
extern GLfloat    zpos;       //Position Int - Out
extern GLfloat    ypos;       //Position Up -Down
extern GLfloat    xpos;       //Position Right -Left
```

Για να δημιουργήσουμε έναν κύβο, πρέπει να μετακινήσουμε το GL matrix στην θέση που επιθυμούμε να ξεκινήσουμε την σχεδίαση του κύβου. Το σημείο αυτό, μετά την δημιουργία, ερμηνεύεται ως το κέντρο του. Η φιλοσοφία σε αυτήν την λειτουργία είναι ίδια με αυτήν της OpenAL, επομένως τις ίδιες τιμές θέτουμε και στους Sources, ώστε να τους έχουμε τοποθετημένους στο κέντρο του κάθε κύβου. Υπενθυμίζουμε ότι οι τιμές αυτές για την OpenAL θα τεθούν κατά το ξεκίνημα του προγράμματος από την συνάρτηση SoundPlay(), ενώ οι τιμές για την GL θα τοποθετούνται κάθε φορά που τρέχει η συνάρτηση DrawGLScen, μιας και τα γραφικά δεν μπορούν να γίνουν «στατικά», αλλά επανασχεδιάζονται κάθε φορά που εκτελείται η συνάρτηση.

```
//Cube Position
GLfloat     cube1x = -3.0f, cube1y = -2.0f,cube1z = -10.0f;
GLfloat     cube2x =  3.0f, cube2y = -2.0f,cube2z = -10.0f;
GLfloat     cube3x = -3.0f, cube3y =  2.0f,cube3z = -10.0f;
GLfloat     cube4x =  3.0f, cube4y =  2.0f,cube4z = -10.0f;
```

```
//source Position
ALfloat Spos1[] = { cube1x, cube1y, cube1z };
ALfloat Spos2[] = { cube2x, cube2y, cube2z };
ALfloat Spos3[] = { cube3x, cube3y, cube3z };
ALfloat Spos4[] = { cube4x, cube4y, cube4z };
```

Στην συνέχεια θέτουμε τις τιμές για την ηχητική geodata. Ο όρος geodata χρησιμοποιείται από τους προγραμματιστές παιχνιδιών και αναφέρεται σε όλες τις συντεταγμένες (x,y,z) που έχουν αλληλεπίδραση με τον χρήστη και υποδηλώνουν συνήθως περιορισμούς. Παραδείγματος χάρη, ένας τοίχος που δεν πρέπει να είναι διαπερατός από ένα εικονικό χαρακτήρα, ορίζεται μέσα σε βάσεις δεδομένων που στο σύνολο τους ονομάζονται geodata.

Στο συγκεκριμένο παράδειγμα χρησιμοποιούνται τέτοιου τύπου τιμές που ερμηνεύουν τα «τοιχώματα» του κάθε κύβου. Σε αυτό το σημείο πρέπει να αναφέρουμε ότι, η κάθε πλευρά του κύβου είναι τοποθετημένη σε απόσταση ίση με 1 από το κεντρικό σημείο του κύβου, δηλαδή ο κύβος-1 που είναι τοποθετημένος στο σημείο (χ = -3.0f , y = -2.0f, z = -10.0f) για τον άξονα χ, θα διαθέτει δύο πλευρές που η αριστερή του θα έχει την τιμή -4.0f και η δεξιά θα είναι στο σημείο -2.0f. Αντίστοιχα μπορούμε να εξάγουμε όλες τις πλευρές του κάθε κύβου. Οι τιμές αυτές μας δίνουν τις απαραίτητες πληροφορίες που χρειαζόμαστε, ώστε να αλλάζει ο ήχος κάθε φορά που εισερχόμαστε στον κάθε κύβο. Για παράδειγμα, αν ο χρήστης μπει στον άξονα χ μέσα στα σημεία -4.0f έως -2.0f, είναι μέσα στον κύβο-1 και επομένως πρέπει να δημιουργηθεί η ηχητική αλλαγή που ακούγεται στο παράδειγμα. Οι τιμές αυτές για τον κάθε κύβο ξεχωριστά έχουν χωριστεί σε min και max, όπου min είναι η αριστερή πλευρά του κύβου. Ιδιαίτερη προσοχή χρειάζεται στα πρόσημα, μιας οι τιμές αυτές δεν αποτελούν μέγιστη και ελάχιστη τιμή, αλλά τις δύο πλευρές.

```
//Sound Geodata for Enviroment
GLfloat     cube1xmin =  -4.0f, cube1xmax =  -2.0f;
GLfloat     cube1ymin =  -3.0f, cube1ymax =  -1.0f;
GLfloat     cube1zmin = -11.0f, cube1zmax =  -9.0f;

GLfloat     cube2xmin =   4.0f, cube2xmax =   2.0f;
GLfloat     cube2ymin =  -3.0f, cube2ymax =  -1.0f;
GLfloat     cube2zmin = -11.0f, cube2zmax =  -9.0f;

GLfloat     cube3xmin =  -4.0f, cube3xmax =  -2.0f;
GLfloat     cube3ymin =   3.0f, cube3ymax =   1.0f;
GLfloat     cube3zmin = -11.0f, cube3zmax =  -9.0f;

GLfloat     cube4xmin =   4.0f, cube4xmax =   2.0f;
GLfloat     cube4ymin =   3.0f, cube4ymax =   1.0f;
GLfloat     cube4zmin = -11.0f, cube4zmax =  -9.0f;
```

Εντέλει ορίζονται και τέσσερεις flag που ενεργοποιούνται όταν ο χρήστης βρεθεί εντός των έξι πλευρών:

```
//Lisener is into cube 1 (TRUE or FALSE flag)
ALboolean         LisenerCube1      = AL_FALSE;
//Lisener is into cube 2 (TRUE or FALSE flag)
ALboolean         LisenerCube2      = AL_FALSE;
//Lisener is into cube 3 (TRUE or FALSE flag)
ALboolean         LisenerCube3      = AL_FALSE;
//Lisener is into cube 4 (TRUE or FALSE flag)
ALboolean         LisenerCube4      = AL_FALSE;
```

Κάθε φορά που εκτελείται η συνάρτηση DrawGLScen, εκκαθαρίζονται οι buffers της GL και στην συνέχεια τοποθετείται το Matrix στην αρχική του θέση, όπου είναι το κέντρο της οθόνης. Στην συνέχεια τοποθετούνται οι τιμές x, y, z που εξάγονται κάθε φορά από την ρουτίνα done στον listener και ταυτόχρονα, με της ίδιες τιμές, αλλά αντεστραμμένες, αναγκάζουμε το Matrix να δημιουργήσει την ψευδαίσθηση της κίνησης.

```
GLboolean DrawGLScene()//All  Drawing System
{

GLboolean         bReturn = GL_TRUE;
ALfloat ListenerPos[] = {xpos,ypos,zpos}; //Position of the listener

// Clear The Screen And The Depth Buffer
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
//Replace The Current Matrix With The Identity Matrix
glLoadIdentity();

      glTranslatef(-xpos,-ypos,-zpos);//Move the player
      alListenerfv(AL_POSITION,ListenerPos);//Move the player
```

Πρέπει να σημειωθεί ότι η συγκεκριμένη λειτουργία που εκτελείται δεν αποτελεί πραγματική κίνηση μιας κάμερας GL, αλλά έναν εύκολο τρόπο να κινηθούμε γραφικά μέσα στην Οθόνη, δημιουργώντας την ψευδαίσθηση της κίνησης. Όπως έχει προαναφερθεί, ο χειρισμός της κάμερας GL - Matrix αποτελεί μια ιδιαίτερα δύσκολη λειτουργία.

Αφού έχουμε λάβει τις τιμές για την θέση του Listener εισάγουμε τον έλεγχο της ηχητικής geodata. Για τον κύβο-1 ο κώδικας έχει ως εξής:

```
//AL Geodata for Sound Enviroment

//Cube1
if(xpos >= cube1xmin && xpos <= cube1xmax && ypos >= cube1ymin
   && ypos <= cube1ymax &&zpos >= cube1zmin && zpos <= cube1zmax)
{
      if(LisenerCube1 == AL_FALSE){
            LisenerCube1 = AL_TRUE;//Lisener is into cube
            alSource3i(Sources[0], AL_AUXILIARY_SEND_FILTER,
                        EffectSlot, 0, AL_FILTER_NULL);
            alSourcei(Sources[1], AL_DIRECT_FILTER, Filter);
            alSourcei(Sources[2], AL_DIRECT_FILTER, Filter);
            alSourcei(Sources[3], AL_DIRECT_FILTER, Filter);
            }
      }
else if (LisenerCube1 == AL_TRUE){//Lisener is not into cube  now
      LisenerCube1 = AL_FALSE;//Lisener is out cube
      alSource3i(Sources[0], AL_AUXILIARY_SEND_FILTER,
                  AL_EFFECTSLOT_NULL, 0, AL_FILTER_NULL);
      alSourcei(Sources[1], AL_DIRECT_FILTER, AL_FILTER_NULL);
      alSourcei(Sources[2], AL_DIRECT_FILTER, AL_FILTER_NULL);
      alSourcei(Sources[3], AL_DIRECT_FILTER, AL_FILTER_NULL);
}
```

Όπως φαίνεται, παίρνουμε τις τιμές που έχουμε τοποθετήσει και στον listener και ελέγχουμε για κάθε άξονα αν ο listener βρίσκεται εντός των δύο τιμών. Αν ο listener είναι εντός των δύο τιμών για κάθε άξονα, τότε είναι μέσα στον κύβο. Στην συνέχεια ελέγχουμε να δούμε αν ο listener ήταν και την προηγούμενη φορά που εκτελέστηκε η συνάρτηση μέσα στο κύβο, μέσω της flag Liseneregube1. Αν δεν βρισκόταν μέσα, πρέπει να αλλάξουμε το ηχητικό περιβάλλον του χρήστη. Αυτό επιτυγχάνεται τοποθετώντας στον source-0 που βρίσκεται στο κέντρο του κύβου σε ένα auxiliary effect slot, που κατά το ξεκίνημα της εφαρμογής (συνάρτηση SoundPlay())του έχουμε τοποθετήσει ένα reverb με decay time 20.0f

```
alEffecti( Effect, AL_EFFECT_TYPE, AL_EFFECT_REVERB );
alEffectf(Effect, AL_REVERB_DECAY_TIME, 20.0f);
alAuxiliaryEffectSloti( EffectSlot, AL_EFFECTSLOT_EFFECT, Effect );
```

Αντίστοιχα όλοι οι άλλοι Sources που είναι έξω από τον κύβο, τοποθετούνται σε αρκετά ισχυρό φίλτρο που ελαττώνει και την ολική ένταση, αλλά και τις υψηλές συχνότητες. Και το φίλτρο αυτό, έχει ενεργοποιηθεί κατά την εκκίνηση της εφαρμογής.

```
alFilteri(Filter, AL_FILTER_TYPE, AL_FILTER_LOWPASS);
alFilterf(Filter, AL_LOWPASS_GAIN, 0.1f);
alFilterf(Filter, AL_LOWPASS_GAINHF, 0.1f);
```

Υπενθυμίζουμε ότι, κατά το ξεκίνημα της εφαρμογής (συνάρτηση SoundPlay()), είχαμε ελέγξει όλη την διαδικασία που εκτελείται στην ηχητική geodata, ώστε να μπορούμε σε αυτό το σημείο να αποσυνδέουμε και να συνδέουμε τους Sources, χωρίς να χρειάζεται να κάνουμε έλεγχο για σφάλματα.

Εντέλει, όταν ο Listener βγει από τον κύβο (δηλαδή η πρώτη if είναι ψευδής), αλλά η flag συνεχίζει να είναι TRUE, σημαίνει ότι ο χρήστης, μόλις την συγκεκριμένη στιγμή βγήκε από τον κύβο. Η flag τίθεται FALSE (δηλαδή ο χρήστης είναι πλέον έξω από τον κύβο) και όλοι οι Sources αποσυνδέονται από τα αντίστοιχα objects που είχαν συνδεθεί όταν ο χρήστης μπήκε μέσα στον κύβο.

Η ίδια διαδικασία πραγματοποιείται και για τους άλλους τρεις κύβους, με την διαφορά ότι ελέγχεται διαφορετική ηχητική geodata και flag, ενώ κάθε φορά επηρεάζονται οι sources που είναι έξω από τον συγκεκριμένο κύβο με low pass filters και ο source που είναι μέσα με auxiliary reverb.

Μετά την ολοκλήρωση της συγκεκριμένης διαδικασίας, αναγκάζουμε το matrix να κινηθεί, ώστε να δημιουργήσει τους κύβους. Ουσιαστικά χρησιμοποιούμε τις εντολές push και pop matrix, που αναγκάζουν το matrix να εκτελέσει τις εντολές που περιέχονται μέσα σε αυτές και στην συνέχεια να επανέλθει στην προηγούμενη κατάστασή του, που ήταν το σημείο που έβλεπε ο χρήστης. Επομένως, τοποθετούμε το matrix στο σημείο που

έχουμε ορίσει ότι θα είναι το κέντρο του κύβου, του λέμε ότι θα χρησιμοποιήσει το αντίστοιχο 2D textures και τέλος μπαίνουμε μέσα στην συνάρτηση drawing quads, που δίνονται οι εντολές για την σχεδίαση του κύβου. Οι εντολές αυτές είναι αντίστοιχες με εκείνες της σχεδίασης ενός τετραγώνου και αναλύθηκαν στο παράδειγμα οκτώ, με την διαφορά ότι σχεδιάζουμε έξι τετράγωνα που αποτελούν ένα κύβο. Αυτή η διαδικασία πραγματοποιείται για την δημιουργία και των τεσσάρων κύβων αντίστοιχα.

```
glPushMatrix();
      glTranslatef( cube1x,cube1y,cube1z);//Move Into The Screen
      glBindTexture(GL_TEXTURE_2D, texture[0]);//Get tetxure
      DrawingQuads();//Drawing Quads
glPopMatrix();
```

Εντέλει, αφού ολοκληρωθούν όλες αυτές οι διαδικασίες και έλεγχοι, το πρόγραμμα επιστρέφει στην ρουτίνα done και την επανεκτελεί, μέχρις ότου να λάβει ένα μήνυμα εξόδου. Όταν πραγματοποιηθεί η έξοδος από την ρουτίνα, εκτελούνται τέσσερις συναρτήσεις πριν να κλείσει το πρόγραμμα.

```
/* ================================================================
/*                       Exit -Close System
/* ================================================================
      ExitPlay();
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitGLWindow();//Close Window -OpenAL
return (msg.wParam);//Exit The Program.Return PostQuitMessage(0);
}
```

Η συνάρτηση ExitGLWindows κλείνει την OpenGL και το δημιουργημένο παράθυρο. Οι συναρτήσεις CloseALUT και CloseAL κλείνουν τα αντίστοιχα API, όπως έχει περιγραφεί στα προηγούμενα παραδείγματα. Η συνάρτηση ExitPlay αποδεσμεύει και κλείνει όλα τα Objects, που δημιουργήθηκαν με την SoundPlay.

Ο κώδικας της συγκεκριμένης συνάρτησης φαίνεται παρακάτω:

```
ALvoid ExitPlay()
{
alAuxiliaryEffectSlotf(EffectSlot, AL_EFFECTSLOT_GAIN, 0.1f);
alSource3i(Sources[0],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
           AL_FILTER_NULL);
alSource3i(Sources[1],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
           AL_FILTER_NULL);
alSource3i(Sources[2],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
           AL_FILTER_NULL);
alSource3i(Sources[3],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
           AL_FILTER_NULL);
```

```
alSourcef(Sources[0],AL_GAIN,0.1f);
alSourcef(Sources[1],AL_GAIN,0.1f);
alSourcef(Sources[2],AL_GAIN,0.1f);
alSourcef(Sources[3],AL_GAIN,0.1f);

alSourcei(Sources[0], AL_DIRECT_FILTER, AL_FILTER_NULL);
alSourcei(Sources[1], AL_DIRECT_FILTER, AL_FILTER_NULL);
alSourcei(Sources[2], AL_DIRECT_FILTER, AL_FILTER_NULL);
alSourcei(Sources[3], AL_DIRECT_FILTER, AL_FILTER_NULL);

alutSleep(3);

alSourceStop(Sources[0]);
alSourceStop(Sources[1]);
alSourceStop(Sources[2]);
alSourceStop(Sources[3]);

alutSleep(3);

alFilteri(Filter, AL_FILTER_TYPE, AL_FILTER_NULL);
alEffecti( Effect, AL_EFFECT_TYPE, AL_FILTER_NULL );

alDeleteAuxiliaryEffectSlots(1, &EffectSlot);
alDeleteFilters(1, &Filter);
alDeleteEffects(1, &Effect);
alDeleteSources(4,Sources);
alDeleteBuffers(4,Buffer);
}
```

Ο λόγος που πραγματοποιείται αυτή η διαφοροποίηση της συγκεκριμένης λειτουργίας από τα προηγούμενα παραδείγματα, έχει άμεση σχέση με την χρήση του reverb και την τιμή decay time 20.0f. Αν δεν τοποθετηθούν οι τιμές gain σε τόσο χαμηλές τιμές και δεν δώσουμε τον χρόνο ώστε να κλείσουνε τελείως οι ανακλάσεις (που μπορεί να μην είναι αντιληπτές από ένα σημείο και μετά, αλλά θα προκαλέσουν ένα κλικ κατά το κλείσιμο της εφαρμογής), τότε όταν ο χρήστης βρίσκεται μέσα σε ένα κύβο και κλείνει την εφαρμογή, θα ακούει ένα πολύ δυσάρεστο κλικ. Στο συγκεκριμένο παράδειγμα, αυτό μπορεί να επιτευχθεί μόνο όταν ο χρήστης κλείσει την εφαρμογή από την καρτέλα που παρουσιάζεται όταν πατήσει τα πλήκτρα ctrl-alt-delete και επιλέξει από την καρτέλα διεργασίες το «τέλος διεργασίας».

Με οποιονδήποτε άλλο τρόπο και αν κλείσει η εφαρμογή, θα περάσει μέσα από την συγκεκριμένη συνάρτηση και θα μειώσει σταδιακά το ηχητικό αποτέλεσμα. Κατά την λειτουργία αυτή, βέβαια, δεν είναι απαραίτητο η εφαρμογή να είναι σε sleep mode, όπως έχει τοποθετηθεί στην συγκεκριμένη περίπτωση, αλλά μπορεί να κλείνει τις υπόλοιπες λειτουργίες.

Ουσιαστικά από το σημείο που θα κλείσουν οι ήχοι και μετά, είναι ιδιαίτερα δύσκολο να παρατηρηθεί αυτό το ηχητικό αποτέλεσμα. Για αυτόν τον λόγο η εφαρμογή τίθεται σε sleep mode, ώστε να υπάρχει και ένδειξη από την εικόνα για το τι ακριβώς συμβαίνει και να μπορεί να παρατηρηθεί ο ήχος του reverb, που εξακολουθεί για ελάχιστο χρονικό διάστημα. Επίσης πρέπει να σημειωθεί ότι ο χρόνος αυτός διαφέρει από κάρτα ήχου σε κάρτα ήχου και είναι προτιμότερο να είναι μεγαλύτερος από αυτόν που χρειάζεται κανονικά.

Ολοκληρώνοντας το τελευταίο παράδειγμα της πτυχιακής εργασίας παρατίθεται ο πλήρης κώδικας του συγκεκριμένου παραδείγματος:

/*Include.h*/

```
#ifndef INCLUDE_H_
#define INCLUDE_H_

/* ================================================================ */
/*                       Inlcude  Libraries                         */
/* ================================================================ */

//#pragma comment(lib, "winmm.lib") //Link With Windows MultiMedia
lib
//Link With OpenAL Lib
#pragma comment(lib, "OpenAL32.lib")
#pragma comment(lib, "alut.lib")
#pragma comment(lib, "EFX-Util.lib")

//Link With Microsoft OpenGL lib
#pragma comment (lib, "opengl32.lib")
//Link With Microsoft OpenGL Utility lib
#pragma comment (lib, "glu32.lib")
//Link With Win32 GLAUX lib
#pragma comment (lib, "glaux.lib")
//#pragma comment (lib, "glut32.lib")//Link With GLUT  lib

/* ================================================================ */
/*                Standard Include Libraries Files                  */
/*================================================================= */
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <stdarg.h>
#include <Windows.h>
#include <ctype.h>
#include <math.h>
#include <time.h>
#include <conio.h>
#include <malloc.h>
#include <memory.h>
#include <tchar.h>
#include <iostream>

/* ================================================================ */
/*                Reporting Errors Standard Libraries Files         */
/* ================================================================ */
#include <errno.h> //int errno
#include <assert.h>//assert(Value);

/* ================================================================ */
/*    Inlcude Full Sound OpenlAL Libraries Files                    */
/* ================================================================ */
#include <al.h>
#include <alc.h>
#include <alut.h>
#include <efx.h>
```

```
/* ========================================================== */
/*          Include Full Graphic OpenGL Libraries Files       */
/* ========================================================== */
#include <gl/GL.h>//VC LIB
#include <gl/GLU.h>//VC LIB
#include <gl/GLAux.h>//VC LIB
//#include <gl/GLUT.h>

/* ============================================================ */
/*                Inlcude Program  Libraries    Files           */
/* ========== ================================================= */
#include "SystemWindow.h"  //Window Setup
#include "InitExitAL.h"    //Initializing & exiting OpenAL and ALUT
#include "LoadData.h"      //Load Data
#include "Scene.h"         //Window Scene

/* ========================================================== */
/*                 CALLBACK -Declaration For WndProc          */
/* ========================================================== */
LRESULT     CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

#endif /*MAININCLUDE_H_*/
```

/*InitExitAL.h */

```
#ifndef INITEXITAL_H_
#define INITEXITAL_H_
/*============================================================*/
/*                Initialization & Exit Sound Function        */
/* ===========================================================*/
//AL Sound Full Setup (If No Device Sound Close)*/
ALboolean ALSound ();
ALboolean InitOpenAL();//Initialization OpenAL
ALboolean CloseOpenAL();//Exit-Close OpenAL
ALboolean InitALUT();//Initialization ALUT
ALboolean CloseALUT();//Exit-Close ALUT

#endif /*INITEXITAL_H_*/
```

/*InitExitAL.cpp*/

```
#include "Include.h"

/* ========================================================== */
/*                Initialization & Exit Sound Function        */
/* ========================================================== */

ALboolean ALSound (){//AL Sound Full Setup (If No Device Sound Close)

      if (InitOpenAL() == AL_FALSE ){//Initialization OpenAL
            CloseOpenAL();
            return AL_FALSE;
      }
      else if(InitALUT() == AL_FALSE ){//Initialization ALUT
          CloseALUT();
          CloseOpenAL();
            return AL_FALSE;
      }
```

```
	return AL_TRUE;
}
```

```
ALboolean InitOpenAL()//Initialization OpenAL Manually
{

ALCcontext	*pContext = NULL;
ALCdevice	*pDevice = NULL;
ALboolean	bReturn = AL_FALSE;
ALCboolean	currentcon ;//Test For alcMakeContextCurrent

	pDevice = alcOpenDevice(NULL); //Open default device
	if (pDevice){ //pDevice != NULL
		//creates a context using a default device
		pContext = alcCreateContext(pDevice, NULL);
		if (pContext){//pContext != NULL
		//set active context-Makes a pContext the current context
			currentcon = alcMakeContextCurrent(pContext);
			if (currentcon == ALC_TRUE){
				bReturn = AL_TRUE;
			}
		}
	}

	return bReturn;
}
```

```
ALboolean CloseOpenAL()//Exit-Close OpenAL
{
ALCcontext		*pContext = NULL;
ALCdevice		*pDevice  = NULL;

pContext = alcGetCurrentContext(); //open the current context
//open a context's device pointer
pDevice = alcGetContextsDevice(pContext);
alcMakeContextCurrent(NULL);//makes NULL context the current context
alcDestroyContext(pContext);//destroys a context
alcCloseDevice(pDevice);//closes a device

	return AL_TRUE;
}
```

```
ALboolean  InitALUT()//Initialization ALUT
{
	ALboolean		bReturn = AL_TRUE;


	if (!alutInitWithoutContext( NULL , NULL ))
			bReturn = AL_FALSE;

	return bReturn;
}
```

```
ALboolean CloseALUT()//Exit-Close ALUT
{
	ALboolean				bReturn = AL_TRUE;
	if (!alutExit())
		bReturn = AL_FALSE;

	return bReturn;
}
```

/*LoadData.h*/

```
#ifndef LOADDATA_H_
#define LOADDATA_H_
/* ================================================================ */
/*                       General Function                           */
/*================================================================= */
//Path Finder.Test if Path Exist,If Not return FALSE.
//Sound   Data Path is :SoundPath    : Data\\Sounds\\filename
//Texture Data Path is :TexturesPath : Data\\Textures\\filename
char *DataPath(const char *Path,const char *filename);

/* ================================================================ */
/*    Load GLTextures Data - Definition Variable                    */
/* ================================================================  */
//Texture Path :....Data/Textures/
#define TexturesPath          "Data\\Textures\\"
#define texrure1              "1.bmp"
#define texrure2              "2.bmp"
#define texrure3              "3.bmp"
#define texrure4              "4.bmp"
/* ================================================================ */
/*                 Load GLTextures Data Function                    */
/* ================================================================ */
//Loads A Bitmap Image.If Load Failed Return FALSE
AUX_RGBImageRec *LoadBMP(const char *Filename);
// Load Bitmaps And Convert To Textures
GLuint LoadGLTextures();
/* ================================================================ */
/*           Load Sound Data - Definition Variable                  */
/* ================================================================ */
//Sound Path :....Data/Sounds/
#define SoundPath             "Data\\Sounds\\"
#define Sound1                "1.WAV"
#define Sound2                "2.WAV"
#define Sound3                "3.WAV"
#define Sound4                "4.WAV"


/* ================================================================ */
/*          Load Sound Data Function                                */
/* ================================================================ */
//Load a .WAV sound Be Name.If Load Failed Return AL_FALSE
ALuint      LoadWavSound();

#endif /*LOADDATA_H_*/
```

/*LoadData.cpp*/

```
#include "Include.h"
/* ============================================================= */
/*          General Function                                     */
/* ============================================================= */
char fullPath[MAX_PATH];
char *DataPath(const char *Path,const char *filename){
//Path Finder.Test if Path Exist,If Not return FALSE.
//Sound   Data Path is :SoundPath    : Data\\Sounds\\filename
//Texture Data Path is :TexturesPath : Data\\Textures\\filename

      FILE *File=NULL;// File Handle
      if (!filename||!Path){//Make Sure A Filename and Path Was Given
            return FALSE;//If Not Return FALSE
      }
     //Make spring of full Path
      sprintf_s(fullPath,MAX_PATH,"%s%s",Path,filename);
     //Check If The  full Path File Exists-File=fopen(fullPath,"r");
      fopen_s(&File,fullPath,"r");

      if (File){// IF File Exist?
            fclose(File);//Close The Handle
            return fullPath;
      }
      return FALSE;//If Not Return FALSE
}
```

```
/* ============================================================= */
/*               Load GLTextures Data Function                   */
/* ============================================================= */
// RGB IMAGE INFO:The image height and width MUST be a power of 2.
// The width and height must be at least 64 pixels, and for
// compatability reasons, shouldn't be more than 256 pixels. If the
// image you want to use is not 64, 128 or 256 pixels on the width or
// height, resize it in an art program.
//AUX_RGBImageRec : The record will hold the bitmap width,height,and
//data

AUX_RGBImageRec *LoadBMP(const char *Filename)
{//Loads A Bitmap Image.If Load Failed Return FALSE
      char  *fullPath;//Full Path Texture file

      fullPath = DataPath(TexturesPath,Filename);//Path Finder

      if (fullPath){//If full Path exist
            //Load The Bitmap And Return A Pointer
            return auxDIBImageLoad(fullPath);
      }
      return FALSE;//If Not Return NULL
}

GLuint      texture[4];

GLuint LoadGLTextures()
{// Load Bitmaps And Convert To Textures

int               loop;
int               bReturn = GL_FALSE;
GLenum            ErrorGL;//Save GL Error - GL_NO_ERROR.
```

```
glGetError();//Clear error code


// Create Storage Space For The Texture-Create an image record that
// we can store our bitmap in.
// The record will hold the bitmap width, height, and data.
AUX_RGBImageRec *TextureImage[4];

// Clear the image record just to make sure it's empty.
// Set The Pointer To NULL
// memset Info:
// http://www.cplusplus.com/reference/clibrary/cstring/memset.html
// Sizeof Info:
// http://msdn2.microsoft.com/en-us/library/4s7x1k91(VS.71).aspx
// void is not a type of data in this usage, but indicates the
// absence of data.A void* points at objects of unknown size,
// so pointer arithmetic is not defined on them. In C++ the use
// of 'void*' for addresses of objects of an unknown type is still
// legal.http://www.csci.csusb.edu/dick/samples/c++.glossary.html
memset(TextureImage,0,sizeof(void *)*4);

// Load The Bitmap
TextureImage[0]=LoadBMP(texrure1);
TextureImage[1]=LoadBMP(texrure2);
TextureImage[2]=LoadBMP(texrure3);
TextureImage[3]=LoadBMP(texrure4);

//Check For Errors, If Bitmap's Not Found Quit.
if (TextureImage[0] && TextureImage[1]&& TextureImage[2]&&
TextureImage[3])
{
bReturn = GL_TRUE;// Set The Status To TRUE
glGenTextures(4, &texture[0]);// Create 4 Texture
for (loop=0; loop<4; loop++)  // Loop Through 4 Textures
{
// Create MipMapped Texture
// OpenGL to build a mipmapped texture OpenGL tries to
// build different sized high quality textures. When you
// draw a mipmapped texture to the screen OpenGL will select
// the BEST looking texture from the ones it built (texture
// with the most detail) and draw it to the screen instead
// of resizing the original image (which causes detail loss).
glBindTexture  (GL_TEXTURE_2D, texture[loop]);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,
                GL_LINEAR_MIPMAP_NEAREST);
gluBuild2DMipmaps(GL_TEXTURE_2D, 3, TextureImage[loop]->sizeX,
                  TextureImage[loop]->sizeY, GL_RGB,
                  GL_UNSIGNED_BYTE, TextureImage[loop]->data);
}

ErrorGL=glGetError();//!!!Error ????
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE; //IF Error Set The Status To FALSE

for (loop=0; loop<4; loop++)  // Loop Through 4 Textures
{
      if (TextureImage[loop]) // If Texture Exists
      {
```

```
            if (TextureImage[loop]->data) // If Texture Image Exists
            {
                  // Free The Texture Image Memory
                  free(TextureImage[loop]->data);
            }
                 // Free The Image Structure
                  free(TextureImage[loop]);
                  }
      }
}

return bReturn;
}
```

```
/* ================================================================ */
/*                 Load Sound Data Function                         */
/* ================================================================ */

ALuint Buffer[4];

ALuint  LoadWavSound ()
{ //Load a .WAV sound Be Name.If Load Failed Return AL_FALSE

      ALchar         *fullPath;
      fullPath  = DataPath(SoundPath,Sound1);
      Buffer[0] = alutCreateBufferFromFile ( fullPath);
      fullPath  = DataPath(SoundPath,Sound2);
      Buffer[1] = alutCreateBufferFromFile ( fullPath);
      fullPath  = DataPath(SoundPath,Sound3);
      Buffer[2] = alutCreateBufferFromFile ( fullPath);
      fullPath  = DataPath(SoundPath,Sound4);
      Buffer[3] = alutCreateBufferFromFile ( fullPath);

      //Check For Errors, If Sound's Not Found Quit.
      if (Buffer[0]== AL_NONE||Buffer[1]== AL_NONE ||
          Buffer[2]== AL_NONE ||Buffer[3]== AL_NONE)
      {
            alDeleteBuffers(4,Buffer);//delete Buffers
            return AL_FALSE;
      }

      return AL_TRUE;
}
```

/*Scene.h*/

```
#ifndef SCENE_H_
#define SCENE_H_

//All Drawing System
GLboolean DrawGLScene();
// Draw A Quad
GLvoid DrawingQuads();
//sound sutup and paly
ALboolean SoundPlay();
//Exit Sound
ALvoid ExitPlay();
//EFX Function setup
ALboolean EFXFunctionSetup();
#endif /*SCENE_H_*
```

/*Scene.cpp*/

```
#include "Include.h"

// Effect objects
LPALGENEFFECTS alGenEffects = NULL;
LPALDELETEEFFECTS alDeleteEffects = NULL;
LPALISEFFECT alIsEffect = NULL;
LPALEFFECTI alEffecti = NULL;
LPALEFFECTIV alEffectiv = NULL;
LPALEFFECTF alEffectf = NULL;
LPALEFFECTFV alEffectfv = NULL;
LPALGETEFFECTI alGetEffecti = NULL;
LPALGETEFFECTIV alGetEffectiv = NULL;
LPALGETEFFECTF alGetEffectf = NULL;
LPALGETEFFECTFV alGetEffectfv = NULL;

//Filter objects
LPALGENFILTERS alGenFilters = NULL;
LPALDELETEFILTERS alDeleteFilters = NULL;
LPALISFILTER alIsFilter = NULL;
LPALFILTERI alFilteri = NULL;
LPALFILTERIV alFilteriv = NULL;
LPALFILTERF alFilterf = NULL;
LPALFILTERFV alFilterfv = NULL;
LPALGETFILTERI alGetFilteri = NULL;
LPALGETFILTERIV alGetFilteriv = NULL;
LPALGETFILTERF alGetFilterf = NULL;
LPALGETFILTERFV alGetFilterfv = NULL;

// Auxiliary slot object
LPALGENAUXILIARYEFFECTSLOTS alGenAuxiliaryEffectSlots = NULL;
LPALDELETEAUXILIARYEFFECTSLOTS alDeleteAuxiliaryEffectSlots = NULL;
LPALISAUXILIARYEFFECTSLOT alIsAuxiliaryEffectSlot = NULL;
LPALAUXILIARYEFFECTSLOTI alAuxiliaryEffectSloti = NULL;
LPALAUXILIARYEFFECTSLOTIV alAuxiliaryEffectSlotiv = NULL;
LPALAUXILIARYEFFECTSLOTF alAuxiliaryEffectSlotf = NULL;
LPALAUXILIARYEFFECTSLOTFV alAuxiliaryEffectSlotfv = NULL;
LPALGETAUXILIARYEFFECTSLOTI alGetAuxiliaryEffectSloti = NULL;
LPALGETAUXILIARYEFFECTSLOTIV alGetAuxiliaryEffectSlotiv = NULL;
LPALGETAUXILIARYEFFECTSLOTF alGetAuxiliaryEffectSlotf = NULL;
LPALGETAUXILIARYEFFECTSLOTFV alGetAuxiliaryEffectSlotfv = NULL;
```

```
//EFX
ALuint      Effect;//Effect Object
ALuint      Filter;//Filter Object
ALuint      EffectSlot;//Auxiliary Effect Slots Object

ALuint      Sources[4];

//extern form LoadData.cpp
extern GLuint   texture[4]; //load texture data
extern ALuint   Buffer [4];   //load Sound data

//extern form Main.cpp
extern GLfloat    zpos;       //Position Int - Out
extern GLfloat    ypos;       //Position Up -Down
extern GLfloat    xpos;       //Position Right -Left
```

```
//Cube Position
GLfloat     cube1x = -3.0f, cube1y = -2.0f,cube1z = -10.0f;
GLfloat     cube2x =  3.0f, cube2y = -2.0f,cube2z = -10.0f;
GLfloat     cube3x = -3.0f, cube3y =  2.0f,cube3z = -10.0f;
GLfloat     cube4x =  3.0f, cube4y =  2.0f,cube4z = -10.0f;


//source Position
ALfloat Spos1[] = { cube1x, cube1y, cube1z };
ALfloat Spos2[] = { cube2x, cube2y, cube2z };
ALfloat Spos3[] = { cube3x, cube3y, cube3z };
ALfloat Spos4[] = { cube4x, cube4y, cube4z};


//Sound Geodata for Enviroment
GLfloat     cube1xmin =  -4.0f, cube1xmax =  -2.0f;
GLfloat     cube1ymin =  -3.0f, cube1ymax =  -1.0f;
GLfloat     cube1zmin = -11.0f, cube1zmax =  -9.0f;

GLfloat     cube2xmin =   4.0f, cube2xmax =   2.0f;
GLfloat     cube2ymin =  -3.0f, cube2ymax =  -1.0f;
GLfloat     cube2zmin = -11.0f, cube2zmax =  -9.0f;

GLfloat     cube3xmin =  -4.0f, cube3xmax =  -2.0f;
GLfloat     cube3ymin =   3.0f, cube3ymax =   1.0f;
GLfloat     cube3zmin = -11.0f, cube3zmax =  -9.0f;

GLfloat     cube4xmin =   4.0f, cube4xmax =   2.0f;
GLfloat     cube4ymin =   3.0f, cube4ymax =   1.0f;
GLfloat     cube4zmin = -11.0f, cube4zmax =  -9.0f;

//Lisener is into cube 1 (TRUE or FALSE flag)
ALboolean         LisenerCube1      = AL_FALSE;
//Lisener is into cube 2 (TRUE or FALSE flag)
ALboolean         LisenerCube2      = AL_FALSE;
//Lisener is into cube 3 (TRUE or FALSE flag)
ALboolean         LisenerCube3      = AL_FALSE;
//Lisener is into cube 4 (TRUE or FALSE flag)
ALboolean         LisenerCube4      = AL_FALSE;
```

```
GLboolean DrawGLScene()
{//All  Drawing System

GLboolean         bReturn = GL_TRUE;
ALfloat ListenerPos[] = {xpos,ypos,zpos}; //Position of the listener

// Clear The Screen And The Depth Buffer
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
//Replace The Current Matrix With The Identity Matrix
glLoadIdentity();

glTranslatef(-xpos,-ypos,-zpos);//Move the player
alListenerfv(AL_POSITION,ListenerPos);//Move the player

//AL Geodata for Sound Enviroment
//Cube1
if(xpos >= cube1xmin && xpos <= cube1xmax && ypos >= cube1ymin
   && ypos <= cube1ymax &&zpos >= cube1zmin && zpos <= cube1zmax)
```

```
{
      if(LisenerCube1 == AL_FALSE){
            LisenerCube1 = AL_TRUE;//Lisener is into cube
            alSource3i(Sources[0], AL_AUXILIARY_SEND_FILTER,
                        EffectSlot, 0, AL_FILTER_NULL);
            alSourcei(Sources[1], AL_DIRECT_FILTER, Filter);
            alSourcei(Sources[2], AL_DIRECT_FILTER, Filter);
            alSourcei(Sources[3], AL_DIRECT_FILTER, Filter);
      }
}
else if (LisenerCube1 == AL_TRUE){//Lisener is not into cube  now
      LisenerCube1 = AL_FALSE;//Lisener is out cube
      alSource3i(Sources[0], AL_AUXILIARY_SEND_FILTER,
                  AL_EFFECTSLOT_NULL, 0, AL_FILTER_NULL);
      alSourcei(Sources[1], AL_DIRECT_FILTER, AL_FILTER_NULL);
      alSourcei(Sources[2], AL_DIRECT_FILTER, AL_FILTER_NULL);
      alSourcei(Sources[3], AL_DIRECT_FILTER, AL_FILTER_NULL);
}

//Cube 2
if(xpos <= cube2xmin && xpos >= cube2xmax && ypos >= cube2ymin
   && ypos <= cube2ymax &&zpos >= cube2zmin && zpos <= cube2zmax)
{
      if(LisenerCube2 == AL_FALSE){
            LisenerCube2 = AL_TRUE;//Lisener is into cube
            alSource3i(Sources[1], AL_AUXILIARY_SEND_FILTER,
                        EffectSlot, 0, AL_FILTER_NULL);
            alSourcei(Sources[0], AL_DIRECT_FILTER, Filter);
            alSourcei(Sources[2], AL_DIRECT_FILTER, Filter);
            alSourcei(Sources[3], AL_DIRECT_FILTER, Filter);
      }
}
else if (LisenerCube2 == AL_TRUE){//Lisener is not into cube now
      LisenerCube2 = AL_FALSE;//Lisener is out cube
      alSource3i(Sources[1], AL_AUXILIARY_SEND_FILTER,
                  AL_EFFECTSLOT_NULL, 0, AL_FILTER_NULL);
      alSourcei(Sources[0], AL_DIRECT_FILTER, AL_FILTER_NULL);
      alSourcei(Sources[2], AL_DIRECT_FILTER, AL_FILTER_NULL);
      alSourcei(Sources[3], AL_DIRECT_FILTER, AL_FILTER_NULL);
      }

//Cube 3
if(xpos >= cube3xmin && xpos <= cube3xmax && ypos <= cube3ymin
   && ypos >= cube3ymax &&zpos >= cube3zmin && zpos <= cube3zmax){
      if(LisenerCube3 == AL_FALSE){
            LisenerCube3 = AL_TRUE;//Lisener is into cube
            alSource3i(Sources[2], AL_AUXILIARY_SEND_FILTER,
                        EffectSlot, 0, AL_FILTER_NULL);
            alSourcei( Sources[0], AL_DIRECT_FILTER, Filter);
            alSourcei( Sources[1], AL_DIRECT_FILTER, Filter);
            alSourcei( Sources[3], AL_DIRECT_FILTER, Filter);
      }
}
else if (LisenerCube3 == AL_TRUE){//Lisener is not into cube now
      LisenerCube3 = AL_FALSE;//Lisener is out cube
      alSource3i(Sources[2], AL_AUXILIARY_SEND_FILTER,
                  AL_EFFECTSLOT_NULL, 0, AL_FILTER_NULL);
      alSourcei( Sources[0], AL_DIRECT_FILTER, AL_FILTER_NULL);
      alSourcei( Sources[1], AL_DIRECT_FILTER, AL_FILTER_NULL);
```

```
      alSourcei( Sources[3], AL_DIRECT_FILTER, AL_FILTER_NULL);
}
//Cube 4
if(xpos <= cube4xmin && xpos >= cube4xmax && ypos <= cube4ymin
   && ypos >= cube4ymax &&zpos >= cube4zmin && zpos <= cube4zmax){
   if(LisenerCube4 == AL_FALSE){
            LisenerCube4 = AL_TRUE;//Lisener is into cube
            alSource3i (Sources[3], AL_AUXILIARY_SEND_FILTER,
                        EffectSlot, 0, AL_FILTER_NULL);
            alSourcei( Sources[0], AL_DIRECT_FILTER, Filter);
            alSourcei( Sources[1], AL_DIRECT_FILTER, Filter);
            alSourcei( Sources[2], AL_DIRECT_FILTER, Filter);
      }
}
else if (LisenerCube4 == AL_TRUE){//Lisener is not into cube
            LisenerCube4 = AL_FALSE;//Lisener is out cube
            alSource3i(Sources[3], AL_AUXILIARY_SEND_FILTER,
                       AL_EFFECTSLOT_NULL, 0, AL_FILTER_NULL);
            alSourcei( Sources[0], AL_DIRECT_FILTER, AL_FILTER_NULL);
            alSourcei( Sources[1], AL_DIRECT_FILTER, AL_FILTER_NULL);
            alSourcei( Sources[2], AL_DIRECT_FILTER, AL_FILTER_NULL);
}
//End Code for AL Geodata.

glPushMatrix();
      glTranslatef( cube1x,cube1y,cube1z);//Move Into The Screen
      glBindTexture(GL_TEXTURE_2D, texture[0]);//Get tetxure
      DrawingQuads();//Drawing Quads
glPopMatrix();

glPushMatrix();
      glTranslatef( cube2x,cube2y,cube2z);//Move Into The Screen
      glBindTexture(GL_TEXTURE_2D, texture[1]);//Get tetxure
      DrawingQuads();//Drawing Quads
glPopMatrix();

glPushMatrix();
      glTranslatef( cube3x ,cube3y ,cube3z);//Move Into The Screen
      glBindTexture(GL_TEXTURE_2D, texture[2]);//Get tetxure
      DrawingQuads();//Drawing Quads
glPopMatrix();

glPushMatrix();
      glTranslatef( cube4x,cube4y,cube4z);//Move Into The Screen
      glBindTexture(GL_TEXTURE_2D, texture[3]);//Get tetxure
      DrawingQuads();//Drawing Quads
glPopMatrix();

return bReturn;
}
```

```
GLvoid DrawingQuads()
{

glBegin(GL_QUADS);      // Start Drawing Quads

      // Front Face
      glNormal3f( 0.0f,0.0f,1.0f);//Normal Pointing Towards Viewer
      glTexCoord2f(0.0f, 0.0f); glVertex3f(-1.0f, -1.0f,  1.0f);
```

```
	glTexCoord2f(1.0f, 0.0f); glVertex3f( 1.0f, -1.0f,  1.0f);
	glTexCoord2f(1.0f, 1.0f); glVertex3f( 1.0f,  1.0f,  1.0f);
	glTexCoord2f(0.0f, 1.0f); glVertex3f(-1.0f,  1.0f,  1.0f);

	// Back Face
	glNormal3f( 0.0f,0.0f,-1.0f);//Normal Pointing Away From Viewer
	glTexCoord2f(1.0f, 0.0f); glVertex3f(-1.0f, -1.0f, -1.0f);
	glTexCoord2f(1.0f, 1.0f); glVertex3f(-1.0f,  1.0f, -1.0f);
	glTexCoord2f(0.0f, 1.0f); glVertex3f( 1.0f,  1.0f, -1.0f);
	glTexCoord2f(0.0f, 0.0f); glVertex3f( 1.0f, -1.0f, -1.0f);

	// Top Face
	glNormal3f( 0.0f, 1.0f, 0.0f);// Normal Pointing Up
	glTexCoord2f(0.0f, 1.0f); glVertex3f(-1.0f,  1.0f, -1.0f);
	glTexCoord2f(0.0f, 0.0f); glVertex3f(-1.0f,  1.0f,  1.0f);
	glTexCoord2f(1.0f, 0.0f); glVertex3f( 1.0f,  1.0f,  1.0f);
	glTexCoord2f(1.0f, 1.0f); glVertex3f( 1.0f,  1.0f, -1.0f);

	// Bottom Face
	glNormal3f( 0.0f,-1.0f, 0.0f);// Normal Pointing Down
	glTexCoord2f(1.0f, 1.0f); glVertex3f(-1.0f, -1.0f, -1.0f);
	glTexCoord2f(0.0f, 1.0f); glVertex3f( 1.0f, -1.0f, -1.0f);
	glTexCoord2f(0.0f, 0.0f); glVertex3f( 1.0f, -1.0f,  1.0f);
	glTexCoord2f(1.0f, 0.0f); glVertex3f(-1.0f, -1.0f,  1.0f);

	// Right face
	glNormal3f( 1.0f, 0.0f, 0.0f);// Normal Pointing Right
	glTexCoord2f(1.0f, 0.0f); glVertex3f( 1.0f, -1.0f, -1.0f);
	glTexCoord2f(1.0f, 1.0f); glVertex3f( 1.0f,  1.0f, -1.0f);
	glTexCoord2f(0.0f, 1.0f); glVertex3f( 1.0f,  1.0f,  1.0f);
	glTexCoord2f(0.0f, 0.0f); glVertex3f( 1.0f, -1.0f,  1.0f);

	// Left Face
	glNormal3f(-1.0f, 0.0f, 0.0f);// Normal Pointing Left
	glTexCoord2f(0.0f, 0.0f); glVertex3f(-1.0f, -1.0f, -1.0f);
	glTexCoord2f(1.0f, 0.0f); glVertex3f(-1.0f, -1.0f,  1.0f);
	glTexCoord2f(1.0f, 1.0f); glVertex3f(-1.0f,  1.0f,  1.0f);
	glTexCoord2f(0.0f, 1.0f); glVertex3f(-1.0f,  1.0f, -1.0f);
glEnd();
}
```

```
ALboolean SoundPlay()
{
ALenum      Error;

//setup EFX
if(EFXFunctionSetup() == AL_FALSE)
	return AL_FALSE;

alGetError ();//Clear AL error code

//Generate 4 Source
alGenSources (4,Sources);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
	return AL_FALSE;
//Generate 1 Effect Object
alGenEffects(1, &Effect);
Error = alGetError ();
```

```
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//Generate 1 Filter Object
alGenFilters(1, &Filter);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//Generate 1 Auxiliary Effect Slots
alGenAuxiliaryEffectSlots( 1, &EffectSlot );
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//Attach Buffer to Sources
alSourcei (Sources[0], AL_BUFFER, Buffer[0]);
alSourcei (Sources[1], AL_BUFFER, Buffer[1]);
alSourcei (Sources[2], AL_BUFFER, Buffer[2]);
alSourcei (Sources[3], AL_BUFFER, Buffer[3]);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//Set Sound position
alSourcefv(Sources[0], AL_POSITION, Spos1);
alSourcefv(Sources[1], AL_POSITION, Spos2);
alSourcefv(Sources[2], AL_POSITION, Spos3);
alSourcefv(Sources[3], AL_POSITION, Spos4);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//set sound Loop ON
alSourcei(Sources[0], AL_LOOPING, AL_TRUE);//Loop ON
alSourcei(Sources[1], AL_LOOPING, AL_TRUE);//Loop ON
alSourcei(Sources[2], AL_LOOPING, AL_TRUE);//Loop ON
alSourcei(Sources[3], AL_LOOPING, AL_TRUE);//Loop ON
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//Set Enviroment  Reverb.
alEffecti( Effect, AL_EFFECT_TYPE, AL_EFFECT_REVERB );
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//Set Reverb decay Time Full
alEffectf(Effect, AL_REVERB_DECAY_TIME, 20.0f);

//Set Rever to Auxilariary
alAuxiliaryEffectSloti( EffectSlot, AL_EFFECTSLOT_EFFECT, Effect );
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//Set Enviroment  Filter
alFilteri(Filter, AL_FILTER_TYPE, AL_FILTER_LOWPASS);
Error = alGetError ();
```

```
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//Set Lowpass
alFilterf(Filter, AL_LOWPASS_GAIN, 0.1f);
alFilterf(Filter, AL_LOWPASS_GAINHF, 0.1f);

//Test Code 1 : Set Source into Auxilary
alSource3i(Sources[0],AL_AUXILIARY_SEND_FILTER,EffectSlot,0, Filter);
alSource3i(Sources[1],AL_AUXILIARY_SEND_FILTER,EffectSlot,0, Filter);
alSource3i(Sources[2],AL_AUXILIARY_SEND_FILTER,EffectSlot,0, Filter);
alSource3i(Sources[3],AL_AUXILIARY_SEND_FILTER,EffectSlot,0, Filter);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;


//Test Code 2:if Test1 is ok -Remove Source and test if Remove is ok?
alSource3i(Sources[0],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
           AL_FILTER_NULL);
alSource3i(Sources[1],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
           AL_FILTER_NULL);
alSource3i(Sources[2],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
           AL_FILTER_NULL);
alSource3i(Sources[3],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
           AL_FILTER_NULL);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//Test Code 3: set Filter into Source
alSourcei(Sources[0], AL_DIRECT_FILTER, Filter);
alSourcei(Sources[1], AL_DIRECT_FILTER, Filter);
alSourcei(Sources[2], AL_DIRECT_FILTER, Filter);
alSourcei(Sources[3], AL_DIRECT_FILTER, Filter);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//Test Code 4:if Test3 is ok -Remove Filter and test if Remove is ok?
alSourcei(Sources[0], AL_DIRECT_FILTER, AL_FILTER_NULL);
alSourcei(Sources[1], AL_DIRECT_FILTER, AL_FILTER_NULL);
alSourcei(Sources[2], AL_DIRECT_FILTER, AL_FILTER_NULL);
alSourcei(Sources[3], AL_DIRECT_FILTER, AL_FILTER_NULL);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

//play a  source
alSourcePlay (Sources[0]);
alSourcePlay (Sources[1]);
alSourcePlay (Sources[2]);
alSourcePlay (Sources[3]);
Error = alGetError ();
if ( Error != AL_NO_ERROR )
      return AL_FALSE;

return AL_TRUE;
}
```

```
ALvoid ExitPlay()
{

alAuxiliaryEffectSlotf(EffectSlot, AL_EFFECTSLOT_GAIN, 0.1f);

alSource3i(Sources[0],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
           AL_FILTER_NULL);
alSource3i(Sources[1],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
           AL_FILTER_NULL);
alSource3i(Sources[2],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
           AL_FILTER_NULL);
alSource3i(Sources[3],AL_AUXILIARY_SEND_FILTER,AL_EFFECTSLOT_NULL,0,
           AL_FILTER_NULL);

alSourcef(Sources[0],AL_GAIN,0.1f);
alSourcef(Sources[1],AL_GAIN,0.1f);
alSourcef(Sources[2],AL_GAIN,0.1f);
alSourcef(Sources[3],AL_GAIN,0.1f);
alSourcei(Sources[0], AL_DIRECT_FILTER, AL_FILTER_NULL);
alSourcei(Sources[1], AL_DIRECT_FILTER, AL_FILTER_NULL);
alSourcei(Sources[2], AL_DIRECT_FILTER, AL_FILTER_NULL);
alSourcei(Sources[3], AL_DIRECT_FILTER, AL_FILTER_NULL);
alutSleep(3);

alSourceStop(Sources[0]);
alSourceStop(Sources[1]);
alSourceStop(Sources[2]);
alSourceStop(Sources[3]);
alutSleep(3);

alFilteri(Filter, AL_FILTER_TYPE, AL_FILTER_NULL);
alEffecti( Effect, AL_EFFECT_TYPE, AL_FILTER_NULL );
alDeleteAuxiliaryEffectSlots(1, &EffectSlot);
alDeleteFilters(1, &Filter);
alDeleteEffects(1, &Effect);
alDeleteSources(4,Sources);
alDeleteBuffers(4,Buffer);
}
```

```
ALboolean EFXFunctionSetup()//EFX Function setup
{
ALCdevice  *pDevice = NULL;
ALCcontext *pContext = NULL;
ALboolean   Setup = AL_FALSE;

pContext = alcGetCurrentContext();
pDevice = alcGetContextsDevice(pContext);

if (alcIsExtensionPresent(pDevice, (ALCchar*)ALC_EXT_EFX_NAME))
{
// Get function pointers
alGenEffects = (LPALGENEFFECTS)alGetProcAddress("alGenEffects");
alDeleteEffects = (LPALDELETEEFFECTS
)alGetProcAddress("alDeleteEffects");
alIsEffect = (LPALISEFFECT )alGetProcAddress("alIsEffect");
alEffecti = (LPALEFFECTI)alGetProcAddress("alEffecti");
alEffectiv = (LPALEFFECTIV)alGetProcAddress("alEffectiv");
alEffectf = (LPALEFFECTF)alGetProcAddress("alEffectf");
alEffectfv = (LPALEFFECTFV)alGetProcAddress("alEffectfv");
alGetEffecti = (LPALGETEFFECTI)alGetProcAddress("alGetEffecti");
```

```
alGetEffectiv = (LPALGETEFFECTIV)alGetProcAddress("alGetEffectiv");
alGetEffectf = (LPALGETEFFECTF)alGetProcAddress("alGetEffectf");
alGetEffectfv = (LPALGETEFFECTFV)alGetProcAddress("alGetEffectfv");

alGenFilters = (LPALGENFILTERS)alGetProcAddress("alGenFilters");
alDeleteFilters=(LPALDELETEFILTERS)alGetProcAddress("alDeleteFilters");
alIsFilter = (LPALISFILTER)alGetProcAddress("alIsFilter");
alFilteri = (LPALFILTERI)alGetProcAddress("alFilteri");
alFilteriv = (LPALFILTERIV)alGetProcAddress("alFilteriv");
alFilterf = (LPALFILTERF)alGetProcAddress("alFilterf");
alFilterfv = (LPALFILTERFV)alGetProcAddress("alFilterfv");
alGetFilteri = (LPALGETFILTERI )alGetProcAddress("alGetFilteri");
alGetFilteriv= (LPALGETFILTERIV )alGetProcAddress("alGetFilteriv");
alGetFilterf = (LPALGETFILTERF )alGetProcAddress("alGetFilterf");
alGetFilterfv= (LPALGETFILTERFV )alGetProcAddress("alGetFilterfv");

alGenAuxiliaryEffectSlots = (LPALGENAUXILIARYEFFECTSLOTS)
                          alGetProcAddress("alGenAuxiliaryEffectSlots");
alDeleteAuxiliaryEffectSlots = (LPALDELETEAUXILIARYEFFECTSLOTS)
                        alGetProcAddress("alDeleteAuxiliaryEffectSlots");
alIsAuxiliaryEffectSlot = (LPALISAUXILIARYEFFECTSLOT)
                         alGetProcAddress("alIsAuxiliaryEffectSlot");
alAuxiliaryEffectSloti = (LPALAUXILIARYEFFECTSLOTI)
                          alGetProcAddress("alAuxiliaryEffectSloti");
alAuxiliaryEffectSlotiv = (LPALAUXILIARYEFFECTSLOTIV)
                          alGetProcAddress("alAuxiliaryEffectSlotiv");
alAuxiliaryEffectSlotf = (LPALAUXILIARYEFFECTSLOTF)
                          alGetProcAddress("alAuxiliaryEffectSlotf");
alAuxiliaryEffectSlotfv = (LPALAUXILIARYEFFECTSLOTFV)
                          alGetProcAddress("alAuxiliaryEffectSlotfv");
alGetAuxiliaryEffectSloti = (LPALGETAUXILIARYEFFECTSLOTI)
                          alGetProcAddress("alGetAuxiliaryEffectSloti");
alGetAuxiliaryEffectSlotiv = (LPALGETAUXILIARYEFFECTSLOTIV)
                         alGetProcAddress("alGetAuxiliaryEffectSlotiv");
alGetAuxiliaryEffectSlotf = (LPALGETAUXILIARYEFFECTSLOTF)
                        alGetProcAddress("alGetAuxiliaryEffectSlotf");
alGetAuxiliaryEffectSlotfv = (LPALGETAUXILIARYEFFECTSLOTFV)
                         alGetProcAddress("alGetAuxiliaryEffectSlotfv");

if (alGenEffects &&alDeleteEffects && alIsEffect && alEffecti &&
    alEffectiv && alEffectf &&alEffectfv && alGetEffecti &&
    alGetEffectiv && alGetEffectf && alGetEffectfv && alGenFilters &&
    alDeleteFilters && alIsFilter && alFilteri && alFilteriv &&
    alFilterf && alFilterfv &&alGetFilteri &&a lGetFilteriv &&
    alGetFilterf && alGetFilterfv && alGenAuxiliaryEffectSlots &&
    alDeleteAuxiliaryEffectSlots && alIsAuxiliaryEffectSlot &&
    alAuxiliaryEffectSloti &&alAuxiliaryEffectSlotiv &&
    alAuxiliaryEffectSlotf && alAuxiliaryEffectSlotfv &&
    alGetAuxiliaryEffectSloti && alGetAuxiliaryEffectSlotiv &&
    alGetAuxiliaryEffectSlotf &&alGetAuxiliaryEffectSlotfv)

      Setup = AL_TRUE;
}
if (alcIsExtensionPresent(pDevice, "ALC_EXT_EFX") == AL_FALSE){
            Setup = AL_FALSE;
}

return Setup;
}
```

/*SystemWindow.h*/

```
#ifndef SYSTEMWINDOW_H_
#define SYSTEMWINDOW_H_

//Ran widows - Load Data -All Setup For OpenGL-Initialize AL
//Initialization
BOOL WindowsRAN(char* title, int width, int height, BYTE bits);
/
/Create OpenGL Window
BOOL CreateGLWindow(char* title, int width, int height, BYTE bits);
//Exit - Close OpenGLWindow
GLvoid ExitGLWindow();

//Resize The OpenGL Scene Whenever The Window Has Been Resized.
GLboolean ReSizeGLScene(GLsizei width, GLsizei height);
//Initialization OpenGL
GLboolean InitOpenGL();

#endif /*SYSTEMWINDOW_H_*/
```

/*SystemWindow.cpp*/

```
#include "Include.h"

HGLRC           hRC=NULL;     // Open GL Rendering Context
HDC             hDC=NULL;     // Open GL GDI(Graphics Device
                              //Interface)Device Context
HWND            hWnd=NULL;    // Holds Window Handle
HINSTANCE       hInstance;    // Holds The Instance Of The
Application

BOOL WindowsRAN(char* title, int width, int height, BYTE bits)
{
//Ran widows-Load Data-All Setup For OpenGL- AL Initialize

//Setup AL Sound Full
if (!ALSound ()){

MessageBoxA (NULL, "Δεν βρέθηκε κανένας ελεγκτής ήχου στο σύστημά
σας.\n\n\n" "Βεβαιωθείτε ότι η συσκευή σας δουλεύει κανονικά και
εγκαταστήστε το πιο\n""πρόσφατο οδηγό για τη κάρτα ήχου σας.\n",
NULL,MB_ICONWARNING | MB_TASKMODAL );
      return FALSE;
}

//Sound Loading
if(!LoadWavSound ()){//Load a .WAV sound
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL

MessageBoxA (NULL,"Δεν είναι δυνατή η προσπέλαση των αρχείων
ήχου.\n\n\n" "Η επανεγκατάσταση της εφαρμογής ίσως λύσει το πρόβλημα"
, NULL,MB_ICONWARNING | MB_TASKMODAL );

      return FALSE;
}
```

```
//Setup And Create GL Window
if (!CreateGLWindow(title,width,height,bits)){
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitGLWindow();

MessageBoxA (NULL,"Δεν ειναι δυνατή η δημιουργία ενός window.\n\n\n"
"Βεβαιωθείτε ότι η συσκευή σας δουλεύει κανονικά και εγκαταστήστε τον
πιο\n""πρόσφατο οδηγό για την κάρτα γραφικών σας.\n"
, NULL,MB_ICONWARNING | MB_TASKMODAL );

     return FALSE;
}

//Texture Loading
if (!LoadGLTextures())
{
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitGLWindow();

MessageBoxA (NULL, "Δεν είναι δυνατή η προσπέλαση των αρχείων
εικόνας.\n\n\n""Η επανεγκατάσταση της εφαρμογής ίσως λύσει το
πρόβλημα.\n", NULL,MB_ICONWARNING | MB_TASKMODAL );
            return FALSE;
}

//All Property Setup For OpenGL
if (!InitOpenGL()){
      //!!!ERROR:Setup For OpenGL!!!
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitGLWindow();

MessageBoxA (NULL, "Δεν είναι δυνατή η δημιουργία ενός GL
window.\n\n\n"Βεβαιωθείτε ότι η συσκευή σας δουλεύει κανονικά και
εγκαταστήστε τον πιο\n""πρόσφατο οδηγό για την κάρτα γραφικών σας.\n"
, NULL,MB_ICONWARNING | MB_TASKMODAL );
      return FALSE;
}


if(!SoundPlay()){
      ExitPlay();
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitGLWindow();

MessageBoxA (NULL, "Δεν βρέθηκε κανένας έγκυρος ελεγκτής ήχου στο
σύστημά σας που να υποστηρίζει την εφαρμογή.\n\n\n""Βεβαιωθείτε ότι η
συσκευή σας δουλεύει κανονικά και εγκαταστήστε τον πιο \n""πρόσφατο
οδηγό για τη κάρτα ήχου σας.\n",NULL,MB_ICONWARNING | MB_TASKMODAL );
      return FALSE;
}


return TRUE;
}
```

```
BOOL CreateGLWindow(char* title, int width, int height, BYTE bits){
//This Code Creates  OpenGL Window.  Parameters Are:
//title     - Title To Appear At The Top Of The Window
//width     - Width Of The GL Window Or Fullscreen Mode
//height    - Height Of The GL Window Or Fullscreen Mode
//bits      - Number Of Bits To Use For Color (8/16/24/32)

GLuint PixelFormat;// Holds The Results After Searching For A Match

WNDCLASS     wc;        // Windows Class Structure
DWORD       dwExStyle;  // Window Extended Style
DWORD       dwStyle;    // Window Style

// Grabs Rectangle Upper Left / Lower Right Values
RECT        WindowRect;

WindowRect.left=(long)0;        // Set Left Value To 0
WindowRect.right=(long)width;   // Set Right Value To Requested Width
WindowRect.top=(long)0;         // Set Top Value To 0
WindowRect.bottom=(long)height;//Set Bottom Value To Requested Height

hInstance    = GetModuleHandle(NULL);//Grab An Instance For Our Window

//CS_HREDRAW :Redraws the entire window if a movement or size
//adjustment changes the width of the client area.
//CS_VREDRAW :Redraws the entire window if a movement or size
//adjustment changes the height of the client area.
//CS_OWNDC:Allocates a unique device context for each window in the
//class
//Info:http://msdn2.microsoft.com/en-us/library/ms633574.aspx
wc.style           = CS_HREDRAW | CS_VREDRAW | CS_OWNDC;
wc.lpfnWndProc     = (WNDPROC) WndProc;// WndProc Handles Messages
wc.cbClsExtra      = 0;         // No Extra Window Data
wc.cbWndExtra      = 0;         // No Extra Window Data
wc.hInstance       = hInstance;// Set The Instance
wc.hIcon           = LoadIcon(NULL, IDI_WINLOGO);// Load Default Icon
wc.hCursor         = LoadCursor(NULL, IDC_ARROW);// Load Arrow Pointer
wc.hbrBackground   = NULL;      // No Background Required For GL
wc.lpszMenuName    = NULL;      // We Don't Want A Menu
wc.lpszClassName   = "OpenGL"; // Set The Class Name


if (!RegisterClass(&wc)){//Attempt To Register The Window Class
      ExitGLWindow();
      return FALSE;//!!!ERROR:Failed To Register The Window Class
}

//APPWINDOW:Forces a top-level window onto the taskbar when the
//window is visible.
//WINDOWEDGE:Specifies that a window has a border with a raised edge.
dwExStyle=WS_EX_APPWINDOW | WS_EX_WINDOWEDGE;


//Creates an overlapped window with the WS_OVERLAPPED, WS_CAPTION,
//WS_SYSMENU, WS_THICKFRAME, WS_MINIMIZEBOX, and WS_MAXIMIZEBOX
//styles.
dwStyle=WS_OVERLAPPEDWINDOW;// Windows Style

//AdjustWindowRectEx:Calculates The Required Size Of The Window
```

```
//Rectangle, Based On The Desired Size Of The Client Rectangle. The
//Window Rectangle Can Then Be Passed To The CreateWindowEx Function
//To Create a Window Whose Client Area Is The Desired Size.
//Info:http://msdn2.microsoft.com/en-us/library/ms632667.aspx


if (!AdjustWindowRectEx(&WindowRect, dwStyle, FALSE, dwExStyle)){
      ExitGLWindow();
      return FALSE;//!!!ERROR:Adjust Window To True Requested Size
}


// Create The Window
hWnd=CreateWindowEx(dwExStyle, //Extended Style For The Window
                    "OpenGL",  //Class Name
                     title,    //Window Title
                     dwStyle | //Defined Window Style
                     WS_CLIPSIBLINGS |//Required Window Style
                     WS_CLIPCHILDREN,  //Required Window Style
                     0, 0,             //Window Position
                    //Calculate Window Width
                    WindowRect.right-WindowRect.left,
                    //Calculate Window Height
                    WindowRect.bottom-WindowRect.top,
                    NULL,           //No Parent Window
                    NULL,           //No Menu
                    hInstance,      //Instance
                    NULL);          //Dont Pass Anything To WM_CREATE


if (!hWnd){//!!!ERROR:Window Creation Error.!!!
            ExitGLWindow();
            return FALSE;
}


//PIXELFORMATDESCRIPTOR:Describes The Pixel Format Of A Drawing
//Surface.
//Info:http://msdn2.microsoft.com/en-us/library/ms537569.aspx
//Choose a Format That Supports OpenGL and Double
//Buffering,RGBA(red,green,blue,alpha channel).
//Find a Pixel Format That Matches The Bits we Decided on (16bit,
//24bit, 32bit). Set Up a 16bit Z-Buffer.
//The Remaining Parameters Are Either Not Used Or Are Not Important
//(Aside From The Stencil Buffer And The (Slow) Accumulation Buffer).
//pfd Tells Windows How We Want Things To Be
static      PIXELFORMATDESCRIPTOR pfd=
{
     //Size Of This Pixel Format Descriptor
      sizeof(PIXELFORMATDESCRIPTOR),
      1,                              //Version Number
      PFD_DRAW_TO_WINDOW |            //Format Must Support Window
      PFD_SUPPORT_OPENGL |            //Format Must Support OpenGL
      PFD_DOUBLEBUFFER,               //Must Support Double Buffering
      PFD_TYPE_RGBA,                  //Request An RGBA Format
      bits,                           //Select Our Color Depth
      0, 0, 0, 0, 0, 0,               //Color Bits Ignored
      0,                              //No Alpha Buffer
      0,                              //Shift Bit Ignored
```

```
        0,                              //No Accumulation Buffer
        0, 0, 0, 0,                     //Accumulation Bits Ignored
        bits,                           //16Bit Z-Buffer (Depth Buffer)
        0,                              //No Stencil Buffer
        0,                              //No Auxiliary Buffer
        PFD_MAIN_PLANE,                 //Main Drawing Layer
        0,                              //Reserved
        0, 0, 0                         //Layer Masks Ignored
        };

hDC=GetDC(hWnd);//Get A GL Device Context
if (!hDC){
      ExitGLWindow();
      return FALSE;//!!!ERROR:Can't Create A GL Device Context!!!
}

//ChoosePixelFormat: Attempts To Match An Appropriate Pixel Format
//Supported By a Device Context To a Given Pixel Format
//Specification.
//Info:http://msdn2.microsoft.com/en-us/library/ms537556.aspx
PixelFormat=ChoosePixelFormat(hDC,&pfd);
if (!PixelFormat){
      ExitGLWindow();
      return FALSE;//!!!ERROR:Can't Find A Matching PixelFormat!!!
}

if(!SetPixelFormat(hDC,PixelFormat,&pfd)){//Set The Pixel Format
      ExitGLWindow();
      return FALSE;//!!!ERROR:Can't Set The PixelFormat!!!
}

hRC=wglCreateContext(hDC);//Create A Rendering Context
if (!hRC){
      ExitGLWindow();
      return FALSE;//!!!ERROR:Can't Create A GL Rendering Context!!!
}

if(!wglMakeCurrent(hDC,hRC)){//set active The Rendering Context
      ExitGLWindow();
      return FALSE;//!!!ERROR:Can't Activate The GL Rendering
                   //Context!!!
}

ShowWindow(hWnd,SW_SHOW);//Show The Window
SetForegroundWindow(hWnd);//Slightly Higher Priority
SetFocus(hWnd);//Sets Keyboard Focus To The Window
ReSizeGLScene(width, height);//Set Up Our Perspective GL Screen

return TRUE;
}
```

```
GLvoid ExitGLWindow()
{//Exit - Close Window

HGLRC hRC= wglGetCurrentContext();//Get current GL Rendering Context
HDC   hDC= wglGetCurrentDC();       //Get current GL Device Context


if (hRC){//Rendering Context
```

```
	wglMakeCurrent(NULL,NULL);//makes NULL The DC And RC Contexts
	wglDeleteContext(hRC);//Delete Rendering Context
	hRC=NULL;// Set Rendering Context To NULL
}

if (hDC && !ReleaseDC(hWnd,hDC)){//Release The Device Context
	hDC=NULL;//Set Device Context  NULL
}

if (hWnd && !DestroyWindow(hWnd)){//Destroy The Window
	hWnd=NULL;//Set Window Handle To NULL
}

if (!UnregisterClass("OpenGL",hInstance)){//Unregister Class
	hInstance=NULL;//Set Unregister Class To NULL
}
}
```

```
GLboolean ReSizeGLScene(GLsizei width, GLsizei height)
{

//Resize The OpenGL Scene Whenever The Window Has Been Resized.
//Example:ReSizeGLScene(800,600)

GLenum			ErrorGL;//Save GL Error
GLboolean		bReturn = GL_TRUE;

//Clear error code
//http://msdn2.microsoft.com/en-us/library/ms537109(VS.85).aspx
glGetError();

if (height==0)// Prevent A Divide By Zero By
{
	height=1;// Making Height Equal One
}


glViewport(0,0,width,height);// Reset The Current Viewport
ErrorGL=glGetError();//!!!Error:Failure Reset The Current Viewport!!!
if (ErrorGL  != GL_NO_ERROR)
	bReturn = GL_FALSE;

glMatrixMode(GL_PROJECTION);// Select The Projection Matrix
//GL_PROJECTION:Applies subsequent matrix operations to the
//projection matrix stack.
ErrorGL=glGetError();//!!!Error:Failure Initialize The Matrix State
!!!
if (ErrorGL  != GL_NO_ERROR)
    bReturn = GL_FALSE;


glLoadIdentity();//Replace The Current Matrix With The Identity
Matrix
ErrorGL=glGetError();//!!!Error:Failure Replace Matrix!!!
if (ErrorGL  != GL_NO_ERROR)
    bReturn = GL_FALSE;


// Calculate The Aspect Ratio Of The Window
gluPerspective(45.0f,(GLfloat)width/(GLfloat)height,0.1f,100.0f);
```

```
//set up a perspective  projection matrix
ErrorGL=glGetError();//!!!Error:Failure ......!!!
if (ErrorGL  != GL_NO_ERROR)
     bReturn = GL_FALSE;

glMatrixMode(GL_MODELVIEW);//Select The Modelview Matrix
//GL_MODELVIEW: Applies subsequent matrix operations to
//the modelview matrix stack.
ErrorGL=glGetError();//!!!Error:Failure Initialize The Matrix State
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

glLoadIdentity();//Replace The Current Matrix With The IdentityMatrix
ErrorGL=glGetError();//!!!Error:Failure Replace Matrix!!!
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

return bReturn;

}
```

```
/Red,Blue,Green,Alpha Value

//Ambient Light Values - Without an ambient light, spots where there
//is no diffuse light will appear very dark.
GLfloat LightAmbient[]=  { 0.5f, 0.5f, 0.5f, 1.0f };

//Diffuse Light Values- A diffuse light this bright lights up the
//front of the crate nicely.
GLfloat LightDiffuse[]=  { 1.0f, 1.0f, 1.0f, 1.0f };

//x,y,z,the last number at 1.0f.tells OpenGL the designated
coordinates are the position of the light source.
//Light Position-position the light off the screen,towards the
//viewe(2.0f),
GLfloat LightPosition[]= { 0.0f, 0.0f, 2.0f, 1.0f };
GLboolean InitOpenGL()
{//Initialization  OpenGL

GLenum                  ErrorGL= GL_NO_ERROR;//Save GL Error
GLboolean         bReturn = GL_TRUE;

glGetError();//Clear error code

//Enable Texture Mapping
glEnable(GL_TEXTURE_2D);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

//Enable Smooth Shading
glShadeModel(GL_SMOOTH);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

//Black Background(Red,Green,Blue,Alpha)The Initial Values Are All 0.
glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
ErrorGL=glGetError();
```

```
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

// Depth Buffer Setup. The Initial Value is 1.
glClearDepth(1.0f);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

//Enables Depth Testing
glEnable(GL_DEPTH_TEST);
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

//Specifies The Value Used For Depth-Buffer Comparisons.
//Specifies The Function Used To Compare Each Incoming Pixel z
//Value With The z Value Present In The Depth Buffer. The Comparison
//Is Performed Only If Depth Testing is Enabled.
//GL_LEGUAL:Passes if the incoming z value is less than or equal to
//the stored z value.
//Info:http://msdn2.microsoft.com/en-us/library/ms537051.aspx
glDepthFunc(GL_LEQUAL);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;


//Specify Implementation-Specific Hints
//GL_PERSPECTIVE_CORRECTION_HINT:Indicates the quality of color and
//texture coordinate interpolation.
//GL_NICEST:The most correct,or highest quality,option should be
//chosen.
glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;
//set up the lighting
//Setup The Ambient Light.Set the amount of ambient
//light that light1 will give off.
//Half intensity ambient light
glLightfv(GL_LIGHT1, GL_AMBIENT, LightAmbient);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

//Setup The Diffuse Light.Set up the amount of diffuse
//light that light number one will give off
// Full intensity white light
glLightfv(GL_LIGHT1, GL_DIFFUSE, LightDiffuse);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

// Position The Light.Set the position of the light.2
//unit towards the viewer
glLightfv(GL_LIGHT1, GL_POSITION,LightPosition);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;
```

```
// Enable Light One
glEnable(GL_LIGHT1);
ErrorGL=glGetError();
if (ErrorGL  != GL_NO_ERROR)
      bReturn = GL_FALSE;

return bReturn;
}
```

/*main.cpp*/

```
#include "Include.h"

bool  done=FALSE;        // Bool Variable To Exit Loop - Windows
bool  keys[256];         // Array Used For The Keyboard Routine
bool  active=TRUE;       // Window Active Flag Set To TRUE By Default
MSG   msg;               // Windows Message Structure

GLfloat zpos;                  //Position Int - Out
GLfloat ypos;                  //Position Up -Down
GLfloat xpos;                  //Position Right -Left
```

```
//The WindowProc function is an application-defined function that
//processes messages sent to a window. The WNDPROC type defines a
//pointer to this callback function. WindowProc is a placeholder for
//the application-defined function name.
//Info:http://msdn2.microsoft.com/en-us/library/ms633573.aspx

LRESULT CALLBACK WndProc(HWND  hWnd,      //Handle For This Window
                         UINT  uMsg,      //Message For This Window
                         WPARAM wParam,   //Additional Message Info
                         LPARAM lParam)   //Additional Message Info
{



      switch (uMsg)//Check For Windows Messages
      {
      case WM_ACTIVATE://Watch For Window Activate Message
      {
            if (!HIWORD(wParam)){//Check Minimization State
                  active=TRUE;//Program Is Active
            }
            else{
                  active=FALSE;//Program Is No Longer Active
            }
            return 0;
      }
      //WM_SYSCOMMAND:A window receives this message when the
      //user chooses a command from the Window menu (formerly known
      //as the system or control  menu) or when the user chooses the
      //maximize button, minimize button, restore button, or close
       //button.
      //Info:http://msdn2.microsoft.com/en-us/library/ms646360.aspx
      case WM_SYSCOMMAND://Intercept System Commands
      {
            switch (wParam){//Check System Calls
                  //Screensaver Trying To Start
                  case SC_SCREENSAVE:
                  //Monitor Trying To Enter Powersave.
```

```
                    case SC_MONITORPOWER:
                   // By Returning 0 We Prevent Those Things From
                   // Happening.
                    return 0;
              }
       break;
       }
//Info:http://msdn2.microsoft.com/en-us/library/ms674887.aspx
//Sent As A Signal That A Window Or An Application Should Terminate.
       case WM_CLOSE:
       {//Send A Quit Message.The Variable Done Will Be Set To TRUE,
       //The Main Loop In WinMain() Will Stop, And The Program Will
       //Close.
              PostQuitMessage(0);
              return 0;
       }
//The WM_KEYDOWN message is posted to the window with the keyboard
//focus when a nonsystem key is pressed. A nonsystem key is a key
//that is pressed when the ALT key is not pressed.
//Info:http://msdn2.microsoft.com/en-us/library/ms646280.aspx
       case WM_KEYDOWN:
       {
//If a key is being held down we can find out what key it is by
//reading wParam.We then make that keys cell in the array keys[ ]
//become TRUE.That way I can read the array later on and find out
//which keys are being held down.This allows more than one key to be
//pressed at the same time.
              keys[wParam] = TRUE;// If So, Mark It As TRUE
              return 0;
       }
//Info:http://msdn2.microsoft.com/en-us/library/ms646281.aspx
       case WM_KEYUP:
       {
//If a key has been released we find out which key it was by reading
//wParam. We then make that keys cell in the array keys[] equal
//FALSE.That way when I read the cell for that key I'll know if it's
//still being held down or if it's been released.
//Each key on the keyboard can be represented by a number from 0-255.
//When I press the key that represents the number 40 for example,
//keys[40] will become TRUE. When I let go, it will become FALSE.
              keys[wParam] = FALSE;
              return 0;
       }
       case WM_SIZE://Resize The OpenGL Window
       {

       //LoWord=Width, HiWord=Height
       ReSizeGLScene(LOWORD(lParam),HIWORD(lParam));
              return 0;
       }
       }//END switch (uMsg)

//Pass All Unhandled Messages To DefWindowProc
//Calls the default window procedure to provide default processing
//for any window messages that an application does not process. This
//function ensures that every message is processed.
//Info:http://msdn2.microsoft.com/en-us/library/ms633572.aspx
return DefWindowProc(hWnd,uMsg,wParam,lParam);
}
```

```
int WINAPI WinMain(HINSTANCE  hInstance,   // Instance
                   HINSTANCE  hPrevInstance,// Previous Instance
                   LPSTR      lpCmdLine,  // Command Line Parameters
                   int        nCmdShow)   // Window Show State
{

if (!WindowsRAN("Moving and 3D World",1024,768,32)){
            return 0;//Quit If Window Was Not Created
      }


while(!done)
{

      //Is There A Message Waiting?
      if (PeekMessage(&msg,NULL,0,0,PM_REMOVE)){
      //Have We Received A Quit Message?
            if (msg.message==WM_QUIT){
                  done=TRUE;         //If So done=TRUE
            }
            else{ //If Not, Deal With Window Messages
                  TranslateMessage(&msg); //Translate The Message
                  DispatchMessage(&msg);  //Dispatch The Message
            }
      }
      else{//If There Are No Messages
      // Draw The Scene.  Watch For ESC Key.
            if (active && keys[VK_ESCAPE]){//Quit Received?
                        done=TRUE;  // ESC
             }
            else{//Star DrawGLScene
                // Not Time To Quit, Update Screen
                  DrawGLScene();// Draw The Scene
                 //Get current GL Device Context
                  HDC   hDC= wglGetCurrentDC();
//Specifies a device context. If the current pixel format for the
//window referenced by thisdevice context includes a back buffer,
//the function exchanges the front and back buffers.
//http://msdn2.microsoft.com/en-us/library/ms537551(VS.85).aspx
                  SwapBuffers(hDC); //Swap Buffers (Double Buffering)

                  if (keys[VK_PRIOR]){// Is Page Up Being Pressed
                        ypos+=0.03f;  // If So, Move UP
                  }
                  if (keys[VK_NEXT]){// Is Page Down Being Pressed
                        ypos-=0.03f;// If So, Move Down
                  }
                  if (keys[VK_UP]){ // Is Up Arrow Being Pressed
                        zpos-=0.03f;// If So, Move Into The Screen
                  }
                  if (keys[VK_DOWN]){// Is Down Arrow Being Pressed
                        zpos+=0.03f;// If So, Move Towards The Viewer
                  }
                  if (keys[VK_RIGHT]){// Is Right Arrow Being Pressed
                        xpos+= 0.1f;            // If So, Move Right
                  }
                  if (keys[VK_LEFT]){// Is Left Arrow Being Pressed
                        xpos-= 0.1f;// If So, Move Right
                  }
```

```
            }//End  drawGLScene
      }//End If There Are No Messages
}//End while(!done)
```

```
/* ============================================================ */
/*                Exit -Close System                            */
/* ============================================================ */
      ExitPlay();
      CloseALUT();//Exit-Close ALUT
      CloseOpenAL();//Exit-Close OpenAL
      ExitGLWindow();//Close Window -OpenAL
return (msg.wParam);//Exit The Program.Return PostQuitMessage(0);
}
```

ΠΑΡΑΡΤΗΜΑΤΑ

ΠΑΡΑΡΤΗΜΑ 1 Functions, Primitive Types, Define Values

ΠΙΝΑΚΑΣ 1.1 ALC Functions	
Error Functions	
alcGetError	Ανακτά και εκκαθαρίζει την επικρατούσα κατάσταση λάθους στο context
Context Functions	
alcCreateContext	Δημιουργεί ένα context
alcMakeContextCurrent	Θέτει ένα context ως το τρέχον context
alcProcessContext	Λέει σε ένα context να αρχίσει μια διαδικασία - επεξεργασία
alcSuspendContext	Αναστέλλει την διαδικασία- επεξεργασία σε ένα διευκρινισμένο context
alcDestroyContext	Καταστρέφει ένα context
alcGetCurrentContext	Ανακτά το τρέχον context
alcGetContextsDevice	Ανακτά τον τρέχοντα δείκτη του device context's
Device Functions	
alcOpenDevice	Ανοίγει ένα device
alcCloseDevice	Κλείνει ένα device
Extension Functions	
alcIsExtensionPresent	Ερώτηση εάν μια context extension είναι διαθέσιμη
alcGetProcAddress	Ανακτά τη διεύθυνση μιας context extension function
alcGetEnumValue	Ανακτά την αξία enum για ένα διευκρινισμένο όνομα απαρίθμησης(enumeration)
State Functions	
alcGetString	Επιστρέφει δείκτες String σχετικούς με το context
alcGetIntegerv	Επιστρέφει ακέραιους αριθμούς σχετικούς με το context
Capture Functions *	
alcCaptureOpenDevice	Ανοίγει ένα capture device
alcCaptureCloseDevice	Κλείνει ένα capture device
alcCaptureStart	Αρχίζει μια λειτουργία capture
alcCaptureStop	Σταματά μια λειτουργία capture
alcCaptureSamples	Ολοκληρώνει μια λειτουργία capture, και δεν εμποδίζει.

* Νέες Functions. Προσθήκη στην έκδοση OpenAL 1.1

ΠΙΝΑΚΑΣ 1.2 ALC Primitive Types	
ALCdevice	Define values για τον ορισμό των device
ALCcontext	Define values για τον ορισμό των context
ALCboolean	8-bit boolean
ALCchar	Χαρακτήρας
ALCbyte	Προσημασμένος 8-bit ακέραιος αριθμός συμπληρώματος ως προς 2
ALCubyte	Μη προσημασμένος 8-bit ακέραιος αριθμός
ALCshort	Προσημασμένος 16-bit ακέραιος αριθμός συμπληρώματος ως προς 2
ALCushort	Μη προσημασμένος 16-bit ακέραιος αριθμός
ALCint	Προσημασμένος 32-bit ακέραιος αριθμός συμπληρώματος ως προς 2
ALCuint	Μη προσημασμένος 32-bit ακέραιος αριθμός
ALCsizei	Μη αρνητικό 32-bit δυαδικό μέγεθος ακέραιων αριθμών
ALCenum	Enumerated 32-bit τιμή
ALCfloat	32-bit IEEE754 μεγάλη πραγματική τιμή
ALCdouble	64-bit IEEE754 πολύ μεγάλη πραγματική τιμή
ALCvoid	ALC void τύπος

ΠΙΝΑΚΑΣ 1.3 ALC Define Values	
ALC_INVALID	Άκυρη τιμή(-1)
ALC_FALSE	Boolean False(0)
ALC_TRUE	Boolean True(1)
Context Attributes	
ALC_FREQUENCY	Συχνότητα για τη μίξη του output buffer, σε Hz
ALC_REFRESH	Ανανέωση του διαστήματος, σε Hz
ALC_SYNC	Flag, που δείχνει ένα σύγχρονο context
ALC_MONO_SOURCES	Υποδεικνύει πόσες πηγές μπορούν να υποστηριχθούν από mono data
ALC_STEREO_SOURCES	Υποδεικνύει πόσες πηγές μπορούν να υποστηριχθούν από stereo data
Error Values	
ALC_NO_ERROR	Δεν υπάρχει κανένα ALC τρέχον λάθος
ALC_INVALID_DEVICE	Το device handle ή το όνομα του διευκρινισμένου driver / server είναι άκυρο.(No device)
ALC_INVALID_CONTEXT	Το όρισμα του context δεν είναι μια έγκυρη ταυτότητα context (Invalid context ID)
ALC_INVALID_ENUM	Ένα σύμβολο που χρησιμοποιείται δεν είναι έγκυρο ή είναι μη εφαρμόσιμο. Άκυρη παράμετρος (bad enum)
ALC_INVALID_VALUE	Μια τιμή δεν είναι έγκυρη ή είναι μη εφαρμόσιμη. Άκυρη αξία παραμέτρου enum.(bad enum)
ALC_OUT_OF_MEMORY	Η μνήμη δεν επαρκεί (Out of memory)

String Query	
ALC_DEFAULT_DEVICE_SPECIFIER	Το όνομα του προσδιορισμένου default device
ALC_DEVICE_SPECIFIER	Το όνομα του προσδιορισμένου device ή ένας κατάλογος όλων των διαθέσιμων device
ALC_EXTENSIONS	Ένας κατάλογος των διαθέσιμων context επεκτάσεων (extensions).
ALC_CAPTURE_DEFAULT_DEVICE_SPECIFIER	Το όνομα του προσδιορισμένου default capture device
ALC_CAPTURE_DEVICE_SPECIFIER	Το όνομα του προσδιορισμένου capture device ή ένας κατάλογος όλων των διαθέσιμων capture device
Integer Query	
ALC_ATTRIBUTES_SIZE	Το μέγεθος (ALCint) που απαιτείται για έναν μηδενικό-τελικό κατάλογο χαρακτηριστικών για το τρέχον context. NULL είναι ένα άκυρο device.
ALC_ALL_ATTRIBUTES	Αναμένει έναν προορισμό ALC_ATTRIBUTES_SIZE, και παρέχει έναν κατάλογο ιδιοτήτων για το τρέχον context του προσδιορισμένου device. NULL είναι ένα άκυρο device
ALC_MAJOR_VERSION	Η αναθεώρηση προδιαγραφών για αυτήν την υλοποίηση(major version). NULL είναι ένα αποδεκτό device
ALC_MINOR_VERSION	Η αναθεώρηση προδιαγραφών για αυτήν την υλοποίηση(minor version). NULL είναι ένα αποδεκτό device
ALC_CAPTURE_SAMPLES	Ο διαθέσιμος αριθμός δειγμάτων - capture samples. NULL είναι ένα άκυρο device

ΠΙΝΑΚΑΣ 2.1: AL Functions	
Error Functions	
alGetError	Ανακτά και εκκαθαρίζει την επικρατούσα κατάσταση λάθους AL
Buffer Functions	
alGenBuffers	Δημιουργεί έναν ή περισσότερους buffers
alDeleteBuffers	Διαγράφει έναν ή περισσότερους buffers
alIsBuffer	Εξετάζει εάν ένα όνομα buffer ισχύει
AlGetBufferf	Ανακτά μια floating point παράμετρο ενός buffer
alGetBuffer3f *	Ανακτά τρεις floating point παραμέτρους ενός buffer (v1,v2,v3)
alGetBufferfv	Ανακτά έναν διάνυσμα floating point που είναι παράμετρος ενός buffer
alGetBufferi	Ανακτά μια integer παράμετρο ενός buffer
alGetBuffer3i *	Ανακτά τρεις integer παραμέτρους ενός buffer (v1,v2,v3)
alGetBufferiv	Ανακτά έναν διάνυσμα integer που είναι παράμετρος ενός buffer
alBufferData	Γεμίζει έναν buffer με τα audio αρχεία
alBufferf *	Θέτει μια floating point παράμετρο ενός buffer
alBuffer3f *	Θέτει τρεις floating point παραμέτρους ενός buffer (v1,v2,v3)
alBufferfv *	Θέτει έναν διάνυσμα floating point ως παράμετρο ενός buffer
alBufferi	Θέτει μια integer παράμετρο ενός buffer
alBuffer3i *	Θέτει τρεις integer παραμέτρους ενός buffer (v1,v2,v3)
alBufferiv *	Θέτει έναν διάνυσμα integer ως παράμετρο ενός buffer
Source Functions	
alGenSources	Δημιουργεί μία ή περισσότερες πηγές
alDeleteSources	Διαγράφει μία ή περισσότερες πηγές
alIsSource	Εξετάζει εάν ένα όνομα πηγής ισχύει
alGetSourcef	Ανακτά μια floating point παράμετρο μιας πηγής
alGetSource3f	Ανακτά τρεις floating point που αντιπροσωπεύουν μια παράμετρο μιας πηγής (v1,v2,v3)
alGetSourcefv	Ανακτά έναν διάνυσμα floating point ως παράμετρο μιας πηγής
alGetSourcei	Ανακτά μια integer παράμετρο μιας πηγής
alGetSource3i *	Ανακτά τρεις integer που αντιπροσωπεύουν μια παράμετρο μιας πηγής. (v1,v2,v3)
alGetSourceiv	Ανακτά έναν διάνυσμα integer ως παράμετρο μιας πηγής
alSourcef	Θέτει μια floating point παράμετρο μιας πηγής
alSource3f	Θέτει τρεις floating point παράμετρο μιας πηγής

* Νέες Functions. Προσθήκη στην έκδοση OpenAL 1.1

alSourcefv	Θέτει έναν διάνυσμα floating point ως παράμετρο μιας πηγής
alSourcei	Θέτει μια integer παράμετρο μιας πηγής
alSource3i *	Θέτει τρεις integer παραμέτρους μιας πηγής (v1,v2,v3)
alSourceiv *	Θέτει έναν διάνυσμα integer ως παράμετρο μιας πηγής
alSourcePlay	Αναπαράγει μια πηγή
alSourcePlayv	Αναπαράγει ένα σύνολο πηγών
alSourcePause	Θέτει σε παύση μια πηγή
alSourcePausev	Θέτει σε παύση ένα σύνολο πηγών
alSourceStop	Σταματά μια πηγή
alSourceStopv	Σταματά ένα σύνολο πηγών
alSourceRewind	Σταματά την πηγή και θέτει το status του σε AL_INITIAL
alSourceRewindv	Σταματά ένα σύνολο πηγών και θέτει το status του σε AL_INITIAL
alSourceQueueBuffers	Θέτει ουρά αναμονής σε ένα σύνολο από buffers σε μια πηγή
alSourceUnqueueBuffers	Αφαιρεί τις ουρές αναμονής ενός συνόλου από buffers σε μια πηγή
Listener Functions	
alListenerf	Θέτει μια floating point παράμετρο για τον ακροατή
alListener3f	Θέτει τρεις floating point παραμέτρους για τον ακροατή(v1,v2,v3)
alListenerfv	Θέτει έναν διάνυσμα floating point ως παράμετρο για τον ακροατή
alListeneri	Θέτει μια integer παράμετρο για τον ακροατή
alListener3i *	Θέτει τρεις integer ως παράμετρο για τον ακροατή (v1,v2,v3)
alListeneriv *	Θέτει έναν integer ως παράμετρο για τον ακροατή
alGetListenerf	Ανακτά μια floating point παράμετρο για τον ακροατή
alGetListener3f	Ανακτά τρεις floating point που αντιπροσωπεύουν μια παράμετρο για τον ακροατή (v1,v2,v3)
alGetListenerfv	Ανακτά έναν διάνυσμα floating point ως παράμετρο για τον ακροατή
alGetListeneri	Ανακτά μια integer παράμετρο για τον ακροατή
alGetListener3i	Ανακτά τρεις integer που αντιπροσωπεύουν μια παράμετρο για τον ακροατή (v1,v2,v3)
alGetListeneriv	Ανακτά έναν διάνυσμα integer ως παράμετρο για τον ακροατή
Extension Functions	
alIsExtensionPresent	Εξετάζει εάν μια συγκεκριμένη επέκταση είναι διαθέσιμη για τον οδηγό(driver) OpenAL
alGetProcAddress	Επιστρέφει τη διεύθυνση μιας λειτουργίας επέκτασης OpenAL
alGetEnumValue	Επιστρέφει την τιμή enum που περιγράφεται από ένα string

* Νέες Functions. Προσθήκη στην έκδοση OpenAL 1.1

State Functions	
alEnable	Επιτρέπει μια λειτουργία στον οδηγό(driver)
alDisable	Θέτει εκτός λειτουργίας ένα χαρακτηριστικό γνώρισμα του οδηγού(driver)
alIsEnabled	Επιστρέφει μια boolean ένδειξη εάν ένα συγκεκριμένο χαρακτηριστικό γνώρισμα επιτρέπεται στον οδηγό(driver)
alGetBoolean	Επιστρέφει μια boolean μιας παραμέτρου OpenAL
alGetDouble	Επιστρέφει ένα double, με ακρίβεια προσέγγισης floating point μιας παραμέτρου OpenAL
alGetFloat	Επιστρέφει μια floating point μιας παραμέτρου OpenAL
alGetInteger	Επιστρέφει μια Integer μιας παραμέτρου OpenAL
alGetBooleanv	Επιστρέφει μια boolean μιας παραμέτρου OpenAL
alGetDoublev	Επιστρέφει ένα double δείκτη , με ακρίβεια προσέγγισης floating point μιας παραμέτρου OpenAL
alGetFloatv	Επιστρέφει έναν δείκτη floating point μιας παραμέτρου OpenAL
alGetIntegerv	Επιστρέφει έναν δείκτη Integer μιας παραμέτρου OpenAL
alGetString	Ανακτά μια ιδιότητα OpenAL σε String
alDistanceModel	Επιλέγει το μοντέλο απόστασης OpenAL
alDopplerFactor	Επιλέγει την τιμή του OpenAL Doppler
alSpeedOfSound *	Επιλέγει την ταχύτητα του ήχου για τη χρήση της στον υπολογισμό Doppler
alDopplerVelocity	Επιλέγει την ένταση του ήχου για τη χρήση της στον υπολογισμό Doppler

* Νέες Functions. Προσθήκη στην έκδοση OpenAL 1.1

ΠΙΝΑΚΑΣ 2.2 AL Primitive Types	
ALboolean	8-bit boolean
ALchar	Χαρακτήρας
ALbyte	Προσημασμένος 8-bit ακέραιος αριθμός συμπληρώματος ως προς 2
ALubyte	Μη προσημασμένος 8-bit ακέραιος αριθμός
ALshort	Προσημασμένος 16-bit ακέραιος αριθμός συμπληρώματος ως προς 2
ALushort	Μη προσημασμένος 16-bit ακέραιος αριθμός
ALint	Προσημασμένος 32-bit ακέραιος αριθμός συμπληρώματος ως προς 2
ALuint	Μη προσημασμένος 32-bit ακέραιος αριθμός
ALsizei	Μη αρνητικό 32-bit δυαδικό μέγεθος ακέραιων αριθμών
ALenum	Enumerated 32-bit τιμή
ALfloat	32-bit IEEE754 μεγάλη πραγματική τιμή (floating-point)
ALdouble	64-bit IEEE754 πολύ μεγάλη πραγματική τιμή (floating-point)
ALvoid	ALC void τύπος

ΠΙΝΑΚΑΣ 2.3 AL Define Values	
AL_INVALID	Άκυρη τιμή (-1)
AL_NONE	Μηδενική τιμή (0)
AL_FALSE	Boolean False(0)
AL_TRUE	Boolean True(1)
Error Values	
AL_NO_ERROR	Δεν υπάρχει κανένα AL τρέχον λάθος
AL_INVALID_NAME	Άκυρο όνομα παραμέτρου
AL_ILLEGAL_ENUM	Άκυρη παράμετρος.
AL_INVALID_ENUM	Ένα σύμβολο που χρησιμοποιείται δεν είναι έγκυρο ή είναι μη εφαρμόσιμο. Άκυρη παράμετρος(bad enum)
AL_INVALID_VALUE	Μια αξία δεν είναι έγκυρη ή είναι μη εφαρμόσιμη. Άκυρη αξία παραμέτρου enum.(bad enum)
AL_ILLEGAL_COMMAND	Εσφαλμένη κλήση
AL_INVALID_OPERATION	Εσφαλμένη κλήση
AL_OUT_OF_MEMORY	Η μνήμη δεν επαρκεί (Out of memory)
Numerical Query Definitions - Global tweak age	
AL_DOPPLER_FACTOR	Ανάλυση συνιστωσών για την επίδραση Doppler
AL_DOPPLER_VELOCITY	Παράμετρος του Doppler Velocity
AL_SPEED_OF_SOUND	Ταχύτητα του ήχου στις ίδιες μονάδες με το Velocity
String Queries - Context strings	
AL_VENDOR	Το όνομα του προμηθευτή.
AL_VERSION	Σειρά έκδοσης, της μορφής :"<spec major number>.<spec minor number> <optional vendor version information>"
AL_RENDERER	Πληροφορίες για το συγκεκριμένο Renderer
AL_EXTENSIONS	Ένας κατάλογος των διαθέσιμων επεκτάσεων

Basic Listener and Source Attributes Values	
AL_POSITION	Η τρέχουσα θέση του ακροατή σε τρισδιάστατο χώρο. X, Y, Z.. Default: {0.0f, 0.0f, 0.0f}
AL_VELOCITY	Το τρέχον velocity σε τρισδιάστατο χώρο X, Y, Z.Default: {0.0f, 0.0f, 0.0f }
AL_GAIN	Καθορίζει μια κλιμακωτή διαμόρφωση εύρους. Σαν ιδιότητα πηγής, ισχύει για εκείνη την συγκεκριμένη πηγή μόνο. Σαν ιδιότητα ακροατών, ισχύει αποτελεσματικά για όλες τις πηγές στο τρέχον context. Default:1.0f. Η προεπιλογή 1,0 σημαίνει ότι ο ήχος δεν υφίσταται καμία τροποποίηση .Μια τιμή AL_GAIN 0,5 ισοδυναμεί με μείωση κατά 6 DB ενώ τιμή μηδέν είναι ίση με 0DB.
Listener Attributes Values	
AL_ORIENTATION	Δείχνει τον προσανατολισμό των ακροατών, είναι ένα ζεύγος τριπλών στοιχείων –διανυσμάτων. Default: {(0.0f, 0.0f, 1.0f), (0.0f, 1.0f, 0.0f) }
Source Attributes Values	
AL_SOURCE_STATE	Έκθεση πληροφοριών για την πηγή. Range: AL_INITIAL, AL_PLAYING, AL_PAUSED, AL_STOPPED
AL_INITIAL	Καθαρίζει τη σειρά αναμονής
AL_PLAYING	Ξεκινά την ακρόαση ενός ήχου (playing)
AL_PAUSED	Θέτει σε παύση (pause)
AL_STOPPED	Θέτει εκτός λειτουργίας(stop)
AL_SOURCE_RELATIVE	Αν ενεργοποιηθεί καθορίζει ότι η θέση, η ταχύτητα και οι ιδιότητες κατεύθυνσης μιας πηγής πρόκειται να ερμηνευθούν σχετικά με τη θέση ακροατών. Values: AL_TRUE, AL_FALSE Default: AL_FALSE
AL_LOOPING	Θέτει σε επανεκκίνηση(Looping) Values: AL_TRUE, AL_FALSE Default: AL_FALSE
AL_BUFFER	Διευκρινίζει το τρέχον buffer object
Source Attributes Values: Source type	
AL_SOURCE_TYPE	Είναι μια λειτουργία ανάγνωσης που δείχνει εάν μια πηγή είναι έτοιμη να περιμένει στη σειρά τους buffer, ή είναι έτοιμη να χρησιμοποιήσει έναν static buffer, ή είναι σε μια ακαθόριστη διαδικασία όπου μπορεί να χρησιμοποιηθεί είτε για τη ροή είτε για την αναπαραγωγή ήχου. Values: AL_UNDETERMINED, AL_STATIC, AL_STREAMING Default: AL_UNDETERMINED
AL_STATIC	Η πηγή είναι στατική εάν ένας buffer έχει συνδεθεί χρησιμοποιώντας AL_BUFFER
AL_STREAMING	Η πηγή εκτελεί λειτουργίες εάν ένας ή περισσότεροι απομονωτές έχουν συνδεθεί χρησιμοποιώντας alSourceQueueBuffers
AL_UNDETERMINED	Η πηγή είναι ακαθόριστη όταν συνδέει NULL buffer

Source Attributes Values: Buffer Queue params	
AL_BUFFERS_QUEUED	Εντολή ερώτησης για τον αριθμό των buffers που είναι στη ουρά αναμονής μιας δεδομένης πηγής. Values: [0, any] Default:none
AL_BUFFERS_PROCESSED	Εντολή ερώτησης για τον αριθμό των buffers που έχουν αναπαραχθεί από μια δεδομένη πηγή. Έμμεσα, αυτό δίνει το δείκτη του buffer που παίζει αυτή την περίοδο. Values: [0, any] Default:none
Source Attributes Values: Gain	
AL_MIN_GAIN	Είναι ένα κλιμακωτό κατώτατο όριο εύρους έντασης. Values: [0.0f, 1.0f] Default: 0.0f
AL_MAX_GAIN	Είναι ένα κλιμακωτό ανώτατο όριο εύρους έντασης . Values: [0.0f, 1.0f] Default: 1.0f
Source Attributes Values: Distance Model Attributes	
AL_REFERENCE_DISTANCE	Χρησιμοποιείται για τους υπολογισμούς μείωσης απόστασης ανάλογα με το distance model. Συγκεκριμένη κατώτατη απόσταση αναφοράς οπό την πηγής. Values: [0, any] Default: 1.0f
AL_ROLLOFF_FACTOR	Χρησιμοποιείται για τους υπολογισμούς μείωσης απόστασης. Ειδικός παράγοντας μείωσης της απόστασης πηγής Values: [0, any] Default: 1.0f
AL_MAX_DISTANCE	Δείχνει την μέγιστη απόσταση επάνω από την οποία οι πηγές δεν μειώνονται χρησιμοποιώντας το distance model. Σε αυτήν την περίπτωση, όταν οι αποστάσεις είναι μεγαλύτερες από το AL_MAX_DISTANCE θέτονται ίσες με το AL_MAX_DISTANCE Values: [0, any] Default: 1.0f
Source Attributes Values: Pitch	
AL_PITCH	Διευκρινίζει το pitch που εφαρμόζεται, είτε στην πηγή, είτε στα αποτελέσματα μίξης, στον ακροατή. Values: [0.0f, any] Default: 1.0f
Source Attributes Values: Direction and Cone	
AL_DIRECTION	Διευκρινίζει την τρέχουσα κατεύθυνση. Εάν το AL_DIRECTION δεν είναι ίσο με μηδενικό διάνυσμα, η πηγή είναι κατευθυντική. Default:(0.0f, 0.0f, 0.0f)
AL_CONE_INNER_ANGLE	Εσωτερική γωνία του ηχητικού κώνου σε μοίρες. Η προεπιλογή 360 σημαίνει ότι η εσωτερική γωνία καλύπτει ολόκληρο τον κόσμο και είναι ισοδύναμος με μια παντοκατευθυντική πηγή. Default: 360.0f
AL_CONE_OUTER_ANGLE	Εξωτερική γωνία του ηχητικού κώνου, σε μοίρες. Η προεπιλογή 360 σημαίνει ότι η εξωτερική γωνία καλύπτει ολόκληρο τον κόσμο. Εάν και η εσωτερική γωνία είναι επίσης 360, η ζώνη για την εξαρτημένη μείωση είναι μηδέν. Default: 360.0f

AL_CONE_OUTER_GAIN	Ο παράγοντας με τον οποίο το AL_GAIN πολλαπλασιάζεται για να καθορίσει το αποτελεσματικό κέρδος έξω από τον κώνο, που καθορίζεται από την εξωτερική γωνία. Values: [0.0f, 0.1f] Default: 0.0f
Source Attributes Values: Offset	
AL_SEC_OFFSET	Το offset του ήχου, που εκφράζεται σε δευτερόλεπτα. (η αξία θα επιστρέψει πίσω στο μηδέν για τις πηγές που είναι σε loop). Values: [0, any] Default: none
AL_SAMPLE_OFFSET	Το offset του ήχου, που εκφράζεται σε δείγματα -sample, (η αξία θα επιστρέψει πίσω στο μηδέν για τις πηγές που είναι σε loop). Values: [0, any] Default: none
AL_BYTE_OFFSET	Το offset του ήχου, που εκφράζεται σε ψηφία –byte (η αξία θα επιστρέψει πίσω στο μηδέν για τις πηγές που είναι σε loop). Values: [0, any] Default: none
Buffer Attributes Values	
AL_FREQUENCY	Συχνότητα, δειγμάτων ανά δευτερόλεπτο σε Hertz [Hz], δηλ.Συχνότητα δειγματοληψίας
AL_BITS	Ο αριθμός των bits ανά δείγμα για τα στοιχεία που περιλαμβάνονται στον buffer.
AL_CHANNELS	Ο αριθμός καναλιών για τα στοιχεία που περιλαμβάνονται στον buffer.
AL_SIZE	Μέγεθος των bytes που τοποθετούνται σε έναν buffer
AL_FORMAT_MONO8	Format ήχου – mono 8
AL_FORMAT_MONO16	Format ήχου – mono 16
AL_FORMAT_STEREO8	Format ήχου – stereo 8
AL_FORMAT_STEREO16	Format ήχου – stereo 16
AL_UNUSED	Δήλωση buffer-δεν υποστηρίζεται για δημόσια χρήση ακόμα.
AL_PENDING	Δήλωση buffer-δεν υποστηρίζεται για δημόσια χρήση ακόμα.
AL_PROCESSED	Δήλωση buffer-δεν υποστηρίζεται για δημόσια χρήση ακόμα.
Distance models	
AL_DISTANCE_MODEL	Το τρέχον πρότυπο (model) απόστασης
AL_INVERSE_DISTANCE	Πρότυπο αντίστροφης απόστασης (IASIG I3DL2 model)
AL_INVERSE_DISTANCE_CLAMPED	Σταθερό αντίστροφο πρότυπο απόστασης (IASIG I3DL2 model)
AL_LINEAR_DISTANCE	Γραμμικό πρότυπο απόστασης
AL_LINEAR_DISTANCE_CLAMPED	Σταθερό γραμμικό πρότυπο απόστασης
AL_EXPONENT_DISTANCE	Εκθετικό πρότυπο απόστασης
AL_EXPONENT_DISTANCE_CLAMPED	Σταθερό εκθετικό πρότυπο απόστασης
AL_CHANNEL_MASK	Καθορίζει το masking των καναλιών. Values: [0 – 255]

ΠΙΝΑΚΑΣ 3.1: EFX Functions	
Effect object functions	
alGenEffects	Χρησιμοποιείται για να δημιουργήσει ένα ή περισσότερα Effect objects.
alDeleteEffects	Χρησιμοποιείται για να διαγράψει και να ελευθερώσει τους πόρους του Effect object που δημιουργήθηκαν με την alGenEffects.
alIsEffect	Καθορίζει εάν ένας προσδιορισμός αντικειμένου είναι ένα έγκυρο Effect object.
alEffecti	Θέτει μια integer παράμετρο ενός Effect object
alEffectiv	Θέτει έναν δείκτη integer ως παράμετρο ενός Effect object
alEffectf	Θέτει μια floating point παράμετρο ενός Effect object
alEffectfv	Θέτει έναν δείκτη floating point ως παράμετρο ενός Effect object
alGetEffecti	Ανακτά μια integer παράμετρο ενός Effect object
alGetEffectiv	Ανακτά έναν δείκτη integer που είναι παράμετρος ενός Effect object
alGetEffectf	Ανακτά μια floating point παράμετρο ενός Effect object
alGetEffectfv	Ανακτά έναν δείκτη floating point που είναι παράμετρος ενός Effect object
Filter object functions	
alGenFilters	Χρησιμοποιείται για να δημιουργήσει ένα ή περισσότερα Filter objects
alDeleteFilters	Χρησιμοποιείται για να διαγράψει και να ελευθερώσει τους πόρους του Filter object που δημιουργήθηκαν με την alGenFilters
alIsFilter	Καθορίζει εάν ένας προσδιορισμός αντικειμένου είναι ένα έγκυρο Filter object
alFilteri	Θέτει μια integer παράμετρο ενός Filter object
alFilteriv	Θέτει έναν δείκτη integer ως παράμετρο ενός Filter object
alFilterf	Θέτει μια floating point ιδιότητα ενός Filter object
alFilterfv	Θέτει έναν δείκτη floating point ως ιδιότητα ενός Filter object
alGetFilteri	Ανακτά μια integer παράμετρο ενός Filter object
alGetFilteriv	Ανακτά έναν δείκτη integer που είναι παράμετρο ενός Filter object
alGetFilterf	Ανακτά μια floating point παράμετρο ενός Filter object
alGetFilterfv	Ανακτά έναν δείκτη floating point που είναι παράμετρος ενός Filter object
Auxiliary Slot object functions	
alGenAuxiliaryEffectSlots	Χρησιμοποιείται για να δημιουργήσει ένα ή περισσότερα Auxiliary Effect Slots
alDeleteAuxiliaryEffectSlots	Χρησιμοποιείται για να διαγράψει και να ελευθερώσει τους πόρους του Auxiliary Effect Slot που δημιουργήθηκαν με την alGenAuxiliaryEffectSlots

alIsAuxiliaryEffectSlot	Καθορίζει εάν ένας προσδιορισμός αντικειμένου είναι ένα έγκυρο Auxiliary Effect Slot.
alAuxiliaryEffectSloti	Θέτει μια integer παράμετρο ενός Auxiliary Effect Slot
alAuxiliaryEffectSlotiv	Θέτει έναν δείκτη integer ως παράμετρο ενός Auxiliary Effect Slot
alAuxiliaryEffectSlotf	Θέτει μια floating point παράμετρο ενός Auxiliary Effect Slot
alAuxiliaryEffectSlotfv	Θέτει έναν δείκτη floating point ως παράμετρο ενός Auxiliary Effect Slot
alGetAuxiliaryEffectSloti	Ανακτά μια integer παράμετρο ενός Auxiliary Effect Slot
alGetAuxiliaryEffectSlotiv	Ανακτά έναν δείκτη integer που είναι παράμετρος ενός Auxiliary Effect Slot
alGetAuxiliaryEffectSlotf	Ανακτά μια floating point παράμετρο ενός Auxiliary Effect Slot
alGetAuxiliaryEffectSlotfv	Ανακτά έναν δείκτη floating point που είναι παράμετρος ενός Auxiliary Effect Slot

ΠΙΝΑΚΑΣ 3.2 EFX Define Values	
ALC_EXT_EFX_NAME	Ερώτηση χρήσης για Effect Extension.Δηλώνει εάν μια διευκρινισμένη επέκταση context είναι διαθέσιμη.
Context definitions	
ALC_EFX_MAJOR_VERSION	Χρησιμοποιείται από την εφαρμογή για να ανακτηθεί ο αριθμός έκδοσης των Effects Extension που υποστηρίζεται από την εφαρμογή OpenAL.
ALC_EFX_MINOR_VERSION	Χρησιμοποιείται από την εφαρμογή, για να ανακτηθεί ο δευτερεύων αριθμός έκδοσης των Effects Extension που υποστηρίζεται από την εφαρμογή OpenAL.
ALC_MAX_AUXILIARY_SENDS	Χρησιμοποιείται στην OpenAL κατά τη διάρκεια της δημιουργίας context (alcCreateContext), για να ζητήσει τον μέγιστο αριθμό Auxiliary Effect Slots σε κάθε πηγή. Δεν είναι εγγυημένο ότι ο επιθυμητός αριθμός θα είναι διαθέσιμος. Έτσι πρέπει να γίνει ερώτηση για αυτήν την διαδικασία και μετά να δημιουργηθεί το context χρησιμοποιώντας alcGetIntergerv. Default:2
Listener definitions	
AL_METERS_PER_UNIT	Διευκρινίζει την σχέση μεταξύ των μονάδων που περνούν στις κλήσεις OpenAL όπως η θέση, η ταχύτητα, η απόσταση αναφοράς, κ.λ.π, και του πραγματικού κόσμου. Για να χρησιμοποιηθεί με την alListener λειτουργία, αυτές οι τιμές πρέπει να είναι μοναδικές και να μην συγκρούονται με άλλες τιμές ακροατών AL. Values:> 0 Default: 2

Source definitions	
AL_DIRECT_FILTER	Χρησιμοποιείται για να εφαρμόσει το φιλτράρισμα στο σωστό direct-path (dry signal) μιας πηγής.Values: Filter ID Default: AL_FILTER_NULL
AL_AUXILIARY_SEND_FILTER	Χρησιμοποιείται για να εγκαταστήσει τις συνδέσεις μεταξύ των πηγών και του Auxiliary Effect Slot Values: Filter ID Default: AL_FILTER_NULL
AL_AIR_ABSORPTION_FACTOR	Είναι ένας πολλαπλασιαστής του ποσού απορρόφησης αέρα (Air Absorption) που εφαρμόζεται στην πηγή. Values: 0 – 10 Default: 0
AL_ROOM_ROLLOFF_FACTOR	Είναι μια από δύο διαθέσιμες μεθόδους των επεκτάσεων effect για την επίδραση των ανακλώμενων του ήχων (πρόωρες αντανακλάσεις και αντήχηση), σύμφωνα με την απόσταση πηγής -ακροατή. Σε αυτήν την περίπτωση, εντούτοις, το Room Rolloff ισχύει μόνο για αυτήν την πηγή ήχου ,και επομένως έχει επιπτώσεις μόνο στον ανακλώμενο ήχο που παράγεται από αυτήν την πηγής.Values: 0 – 10 Default: 0
AL_CONE_OUTER_GAINHF	Ενισχύει την κατευθυντικότητα της συγκεκριμένης πηγής. Values: 0.0 – 1.0 Default: 1.0
AL_DIRECT_FILTER_GAINHF_AUTO	Η άμεση-πορεία της συγκεκριμένης πηγής φιλτράρεται αυτόματα σύμφωνα με τον προσανατολισμό της πηγής που σχετίζεται με τον ακροατή και τον καθορισμό της ιδιότητας πηγών AL_CONE_OUTER_GAINHF Values: AL_TRUE – AL_FALSE Default: AL_TRUE
AL_AUXILIARY_SEND_FILTER_GAIN_AUTO	Η ένταση της συγκεκριμένης πηγής μειώνεται αυτόματα σύμφωνα με την απόσταση πηγής – ακροατή (όπως καθορίζεται από τις παραμέτρους κώνων). Values: AL_TRUE – AL_FALSE Default: AL_TRUE
AL_AUXILIARY_SEND_FILTER_GAINHF_AUTO	Η ένταση της συγκεκριμένης πηγής στις υψηλές συχνότητες θα αλλάζει αυτόματα σύμφωνα με την κατευθυντικότητα πηγής όπως τίθεται από την ιδιότητα AL_CONE_OUTER_GAINHF Values: AL_TRUE – AL_FALSE Default: AL_TRUE

Effect object definitions (Για να χρησιμοποιηθούν με τις alEffect functions)	
Reverb Parameters	
AL_REVERB_DENSITY	Values: [0.0, 1.0] Default: 1.0
AL_REVERB_DIFFUSION	Values: [0.0, 1.0] Default: 1.0
AL_REVERB_GAIN	Values: [0.0, 1.0] Default: 0.32
AL_REVERB_GAINHF	Values: [0.0, 1.0] Default: 0.89
AL_REVERB_DECAY_TIME	Values: [0.1, 20.0] Default: 1.49
AL_REVERB_DECAY_HFRATIO	Values: [0.1, 2.0] Default: 0.83
AL_REVERB_REFLECTIONS_GAIN	Values: [0.0, 3.16] Default: 0.05
AL_REVERB_REFLECTIONS_DELAY	Values: [0.0, 0.3] Default: 0.007
AL_REVERB_LATE_REVERB_GAIN	Values: [0.0,10.0] Default: 1.26
AL_REVERB_LATE_REVERB_DELAY	Values: [0.0, 0.1] Default: 0.011
AL_REVERB_AIR_ABSORPTION_GAINHF	Values: [0.892,1.0] Default: 0.994
AL_REVERB_ROOM_ROLLOFF_FACTOR	Values: [0.0. 10.0] Default: 0.0
AL_REVERB_DECAY_HFLIMIT	Values: [AL_FALSE, AL_TRUE] Default: AL_TRUE
Chorus Parameters	
AL_CHORUS_WAVEFORM	Values:[AL_CHORUS_WAVEFORM_SINUSOID,AL_CHORUS_WAVEFORM_TRIANGLE]
AL_CHORUS_PHASE	Values: [-180, 180] Default: 90
AL_CHORUS_RATE	Values: [0.0, 10.0] Default: 1.1
AL_CHORUS_DEPTH	Values: [0.0, 1.0] Default: 0.1
AL_CHORUS_FEEDBACK	Values: [-1.0, 1.0] Default: 0.25
AL_CHORUS_DELAY	Values: [0.0, 0.016] Default: 0.016
Distortion Parameters	
AL_DISTORTION_EDGE	Values: [0.0, 1.0] Default: 0.2
AL_DISTORTION_GAIN	Values: [0.01, 1.0] Default: 0.05
AL_DISTORTION_LOWPASS_CUTOFF	Values: [80.0, 24000] Default: 8000
AL_DISTORTION_EQCENTER	Values: [80.0, 24000] Default: 3600
AL_DISTORTION_EQBANDWIDTH	Values: [80.0, 24000] Default: 3600
Echo Parameters	
AL_ECHO_DELAY	Values: [0.0, 0.207] Default: 0.1
AL_ECHO_LRDELAY	Values: [0.0, 0.404] Default: 0.1
AL_ECHO_DAMPING	Values: [0.0, 0.99] Default: 0.5
AL_ECHO_FEEDBACK	Values: [0.0, 1.0] Default: 0.5
AL_ECHO_SPREAD	Values: [-1.0, 1.0] Default: -1.0
Flanger Parameters	
AL_FLANGER_WAVEFORM	Values:[AL_FLANGER_WAVEFORM_SINUSOID,AL_FLANGER_WAVEFORM_TRIANGLE]
AL_FLANGER_PHASE	Values: [-180, 180] Default: 0
AL_FLANGER_RATE	Values: [0.0, 10.0] Default: 0.27
AL_FLANGER_DEPTH	Values: [0.0, 10.0] Default: 1.0
AL_FLANGER_FEEDBACK	Values: [-1.0, 1.0] Default: -0.5
AL_FLANGER_DELAY	Values: [0.0, 0.004] Default: 0.002

Frequency Shifter Parameters	
AL_FREQUENCY_SHIFTER_FREQUENCY	Values: [0.0,24000.0] Default:0.0
AL_FREQUENCY_SHIFTER_LEFT_DIRECTION	Values: [0, 2]Default: 0
AL_FREQUENCY_SHIFTER_RIGHT_DIRECTION	Values: [0, 2]Default: 0
Vocal Morphed Parameters	
AL_VOCAL_MORPHER_PHONEMEA	Values: [0, 29]Default: 0
AL_VOCAL_MORPHER_PHONEMEA_COARSE_TUNING	Values: [0, 29]Default: 10
AL_VOCAL_MORPHER_PHONEMEB	Values: [-24,24]Default: 0
AL_VOCAL_MORPHER_PHONEMEB_COARSE_TUNING	Values: [-24,24]Default: 0
AL_VOCAL_MORPHER_WAVEFORM	Values: [0, 2]Default: 0
AL_VOCAL_MORPHER_RATE	Values: [0.0,10.0]Default: 1.41
Pitch shifter Parameters	
AL_PITCH_SHIFTER_COARSE_TUNE	Values: [-12, 12]Default: 12
AL_PITCH_SHIFTER_FINE_TUNE	Values: [-50, 50]Default: 0
Ring modulator Parameters	
AL_RING_MODULATOR_FREQUENCY	Values:[0.0,8000.0] Default:440.0
AL_RING_MODULATOR_HIGHPASS_CUTOFF	Values:[0.0,24000.0] Default:800.0
AL_RING_MODULATOR_WAVEFORM	Values: [0, 2]Default: 0
Autowah Parameters	
AL_AUTOWAH_ATTACK_TIME	Values:[0.0001, 1.0] Default:0.06
AL_AUTOWAH_RELEASE_TIME	Values:[0.0001, 1.0] Default:0.06
AL_AUTOWAH_RESONANCE	Values:[2.0,1000.0] Default:1000.0
AL_AUTOWAH_PEAK_GAIN	Values:[0.00003, 31621.0] Default:11.22
Compressor Parameters	
AL_COMPRESSOR_ONOFF	Values: [0, 1]Default: 1
Equalizer Parameters	
AL_EQUALIZER_LOW_GAIN	Values: [0.126, 7.943] Default:1.0
AL_EQUALIZER_LOW_CUTOFF	Values: [50.0, 800.0] Default: 200.0
AL_EQUALIZER_MID1_GAIN	Values: [0.126, 7.943] Default: 1.0
AL_EQUALIZER_MID1_CENTER	Values: [200.0, 3000.0] Default: 500.0
AL_EQUALIZER_MID1_WIDTH	Values: [0.01, 1.0] Default: 1.0
AL_EQUALIZER_MID2_GAIN	Values: [0.126, 7.943] Default: 1.0

AL_EQUALIZER_MID2_CENTER	Values: [1000.0, 8000.0] Default: 3000.0
AL_EQUALIZER_MID2_WIDTH	Values: [0.01, 1.0] Default: 1.0
AL_EQUALIZER_HIGH_GAIN	Values: [0.126, 7.943] Default: 1.0
AL_EQUALIZER_HIGH_CUTOFF	Values: [4000.0, 16000.0] Default: 6000.0
Effect type	
AL_EFFECT_TYPE	Values: EffectType Enum Default: AL_EFFECT_NULL
AL_EFFECT_FIRST_PARAMETER	0
AL_EFFECT_LAST_PARAMETER	32768
Effect type definitions (Για να χρησιμοποιηθεί με AL_EFFECT_TYPE.)	
AL_EFFECT_NULL	Value: 0x0000
AL_EFFECT_REVERB	Value: 0x0001
AL_EFFECT_CHORUS	Value: 0x0002
AL_EFFECT_DISTORTION	Value: 0x0003
AL_EFFECT_ECHO	Value: 0x0004
AL_EFFECT_FLANGER	Value: 0x0005
AL_EFFECT_FREQUENCY_SHIFTER	Value: 0x0006
AL_EFFECT_VOCAL_MORPHER	Value: 0x0007
AL_EFFECT_PITCH_SHIFTER	Value: 0x0008
AL_EFFECT_RING_MODULATOR	Value: 0x0009
AL_EFFECT_AUTOWAH	Value: 0x000A
AL_EFFECT_COMPRESSOR	Value: 0x000B
AL_EFFECT_EQUALIZER	Value: 0x000C
Auxiliary Slot object definitions(Για να χρησιμοποιηθεί με alAuxiliaryEffectSlot functions)	
AL_EFFECTSLOT_EFFECT	Χρησιμοποιείται για να εγκαταστήσει τις συνδέσεις μεταξύ του Effect Object και του Auxiliary Effect Slot Values: Filter ID Default: AL_EFFECT_NULL
AL_EFFECTSLOT_GAIN	Χρησιμοποιείται για να προσδιοριστή το Gain που θα έχει ένα Auxiliary Effect Slot Values: [0.0 - 1.0] Default: 1.0
AL_EFFECTSLOT_AUXILIARY_SEND_AUTO	Χρησιμοποιείται για να ενεργοποιηθεί ή απενεργοποιηθεί η αυτόματη αποστολή ρυθμίσεις βασισμένες στις φυσικές θέσεις των sources και του listener Values: [AL_FALSE, AL_TRUE] Default: AL_TRUE
AL_EFFECTSLOT_NULL	0

Filter object definitions **(Για να χρησιμοποιηθεί με alFilter functions)**	
Lowpass parameters	
AL_LOWPASS_GAIN	Values: [0.0, 1.0] Default: 1.0
AL_LOWPASS_GAINHF	Values: [0.0, 1.0] Default: 1.0
Highpass Parameters	
AL_HIGHPASS_GAIN	Values: [0.0, 1.0] Default: 1.0
AL_HIGHPASS_GAINLF	Values: [0.0, 1.0] Default: 1.0
Bandpass Parameters	
AL_BANDPASS_GAIN	Values: [0.0, 1.0] Default: 1.0
AL_BANDPASS_GAINLF	Values: [0.0, 1.0] Default: 1.0
AL_BANDPASS_GAINHF	Values: [0.0, 1.0] Default: 1.0
Filter type definitions(Για να χρησιμοποιηθεί με AL_FILTER_TYPE)	
AL_FILTER_NULL	Value: 0x0000
AL_FILTER_LOWPASS	Value: 0x0001
AL_FILTER_HIGHPASS	Value: 0x0002
AL_FILTER_BANDPASS	Value: 0x0003
Filter	
Lowpass filter	
LOWPASS_MIN_GAIN	Default:0.0f
LOWPASS_MAX_GAIN	Default:1.0f
LOWPASS_DEFAULT_GAIN	Default:1.0f
LOWPASS_MIN_GAINHF	Default:0.0f
LOWPASS_MAX_GAINHF	Default:1.0f
LOWPASS_DEFAULT_GAINHF	Default:1.0f
Highpass filter	
HIGHPASS_MIN_GAIN	Default:0.0f
HIGHPASS_MAX_GAIN	Default:1.0f
HIGHPASS_DEFAULT_GAIN	Default:1.0f
HIGHPASS_MIN_GAINLF	Default:0.0f
HIGHPASS_MAX_GAINLF	Default:1.0f
HIGHPASS_DEFAULT_GAINLF	Default:1.0f
Bandpass filter	
BANDPASS_MIN_GAIN	Default:0.0f
BANDPASS_MAX_GAIN	Default:1.0f
BANDPASS_DEFAULT_GAIN	Default:1.0f
BANDPASS_MIN_GAINHF	Default:0.0f
BANDPASS_MAX_GAINHF	Default:1.0f

BANDPASS_DEFAULT_GAINHF	Default:1.0f
BANDPASS_MIN_GAINLF	Default:0.0f
BANDPASS_MAX_GAINLF	Default:1.0f
BANDPASS_DEFAULT_GAINLF	Default:1.0f
Effect parameter structures	
AL reverb effect parameter	
AL_REVERB_MIN_DENSITY	Default:0.0f
AL_REVERB_MAX_DENSITY	Default:1.0f
AL_REVERB_DEFAULT_DENSITY	Default:1.0f
AL_REVERB_MIN_DIFFUSION	Default:0.0f
AL_REVERB_MAX_DIFFUSION	Default:1.0f
AL_REVERB_DEFAULT_DIFFUSION	Default:1.0f
AL_REVERB_MIN_GAIN	Default:0.0f
AL_REVERB_MAX_GAIN	Default:1.0f
AL_REVERB_DEFAULT_GAIN	Default:0.32f
AL_REVERB_MIN_GAINHF	Default:0.0f
AL_REVERB_MAX_GAINHF	Default:1.0f
AL_REVERB_DEFAULT_GAINHF	Default:0.89f
AL_REVERB_MIN_DECAY_TIME	Default:0.1f
AL_REVERB_MAX_DECAY_TIME	Default:20.0f
AL_REVERB_DEFAULT_DECAY_TIME	Default:1.49f
AL_REVERB_MIN_DECAY_HFRATIO	Default:0.1f
AL_REVERB_MAX_DECAY_HFRATIO	Default:2.0f
AL_REVERB_DEFAULT_DECAY_HFRATIO	Default:0.83f
AL_REVERB_MIN_REFLECTIONS_GAIN	Default:0.0f
AL_REVERB_MAX_REFLECTIONS_GAIN	Default:3.16f
AL_REVERB_DEFAULT_REFLECTIONS_GAIN	Default:0.05f
AL_REVERB_MIN_REFLECTIONS_DELAY	Default:0.0f
AL_REVERB_MAX_REFLECTIONS_DELAY	Default:0.3f
AL_REVERB_DEFAULT_REFLECTIONS_DELAY	Default:0.007f
AL_REVERB_MIN_LATE_REVERB_GAIN	Default:0.0f
AL_REVERB_MAX_LATE_REVERB_GAIN	Default:10.0f
AL_REVERB_DEFAULT_LATE_REVERB_GAIN	Default:1.26f
AL_REVERB_MIN_LATE_REVERB_DELAY	Default:0.0f
AL_REVERB_MAX_LATE_REVERB_DELAY	Default:0.1f
AL_REVERB_DEFAULT_LATE_REVERB_DELAY	Default:0.011f

AL_REVERB_MIN_AIR_ABSORPTION_GAINHF	Default:0.892f
AL_REVERB_MAX_AIR_ABSORPTION_GAINHF	Default:1.0f
AL_REVERB_DEFAULT_AIR_ABSORPTION_GAINHF	Default:0.994f
AL_REVERB_MIN_ROOM_ROLLOFF_FACTOR	Default:0.0f
AL_REVERB_MAX_ROOM_ROLLOFF_FACTOR	Default:10.0f
AL_REVERB_DEFAULT_ROOM_ROLLOFF_FACTOR	Default:0.0f
AL_REVERB_MIN_DECAY_HFLIMIT	Default:AL_FALSE
AL_REVERB_MAX_DECAY_HFLIMIT	Default:AL_TRUE
AL_REVERB_DEFAULT_DECAY_HFLIMIT	Default:AL_TRUE
AL chorus effect parameter	
AL_CHORUS_MIN_WAVEFORM	Default:0
AL_CHORUS_MAX_WAVEFORM	Default:1
AL_CHORUS_DEFAULT_WAVEFORM	Default:1
AL_CHORUS_WAVEFORM_SINUSOID	Default:0
AL_CHORUS_WAVEFORM_TRIANGLE	Default:1
AL_CHORUS_MIN_PHASE	Default:(-180)
AL_CHORUS_MAX_PHASE	Default:180
AL_CHORUS_DEFAULT_PHASE	Default:90
AL_CHORUS_MIN_RATE	Default:0.0f
AL_CHORUS_MAX_RATE	Default:10.0f
AL_CHORUS_DEFAULT_RATE	Default:1.1f
AL_CHORUS_MIN_DEPTH	Default:0.0f
AL_CHORUS_MAX_DEPTH	Default:1.0f
AL_CHORUS_DEFAULT_DEPTH	Default:0.1f
AL_CHORUS_MIN_FEEDBACK	Default:(-1.0f)
AL_CHORUS_MAX_FEEDBACK	Default:1.0f
AL_CHORUS_DEFAULT_FEEDBACK	Default:0.25f
AL_CHORUS_MIN_DELAY	Default:0.0f
AL_CHORUS_MAX_DELAY	Default:0.016f
AL_CHORUS_DEFAULT_DELAY	Default:0.016f
AL distortion effect parameter	
AL_DISTORTION_MIN_EDGE	Default:0.0f
AL_DISTORTION_MAX_EDGE	Default:1.0f
AL_DISTORTION_DEFAULT_EDGE	Default:0.2f
AL_DISTORTION_MIN_GAIN	Default:0.01f

AL_DISTORTION_MAX_GAIN	Default:1.0f
AL_DISTORTION_DEFAULT_GAIN	Default:0.05f
AL_DISTORTION_MIN_LOWPASS_CUTOFF	Default:80.0f
AL_DISTORTION_MAX_LOWPASS_CUTOFF	Default:24000.0f
AL_DISTORTION_DEFAULT_LOWPASS_CUTOFF	Default:8000.0f
AL_DISTORTION_MIN_EQCENTER	Default:80.0f
AL_DISTORTION_MAX_EQCENTER	Default:24000.0f
AL_DISTORTION_DEFAULT_EQCENTER	Default:3600.0f
AL_DISTORTION_MIN_EQBANDWIDTH	Default:80.0f
AL_DISTORTION_MAX_EQBANDWIDTH	Default:24000.0f
AL_DISTORTION_DEFAULT_EQBANDWIDTH	Default:3600.0f
AL echo effect parameter	
AL_ECHO_MIN_DELAY	Default:0.0f
AL_ECHO_MAX_DELAY	Default:0.207f
AL_ECHO_DEFAULT_DELAY	Default:0.1f
AL_ECHO_MIN_LRDELAY	Default:0.0f
AL_ECHO_MAX_LRDELAY	Default:0.404f
AL_ECHO_DEFAULT_LRDELAY	Default:0.1f
AL_ECHO_MIN_DAMPING	Default:0.0f
AL_ECHO_MAX_DAMPING	Default:0.99f
AL_ECHO_DEFAULT_DAMPING	Default:0.5f
AL_ECHO_MIN_FEEDBACK	Default:0.0f
AL_ECHO_MAX_FEEDBACK	Default:1.0f
AL_ECHO_DEFAULT_FEEDBACK	Default:0.5f
AL_ECHO_MIN_SPREAD	Default:(-1.0f)
AL_ECHO_MAX_SPREAD	Default:1.0f
AL_ECHO_DEFAULT_SPREAD	Default:(-1.0f)
AL Flanger effect parameter	
AL_FLANGER_MIN_WAVEFORM	Default:0
AL_FLANGER_MAX_WAVEFORM	Default:1
AL_FLANGER_DEFAULT_WAVEFORM	Default:1
AL_FLANGER_WAVEFORM_SINUSOID	Default:0
AL_FLANGER_WAVEFORM_TRIANGLE	Default:1
AL_FLANGER_MIN_PHASE	Default:(-180)
AL_FLANGER_MAX_PHASE	Default:180
AL_FLANGER_DEFAULT_PHASE	Default:0

AL_FLANGER_MIN_RATE	Default:0.0f
AL_FLANGER_MAX_RATE	Default:10.0f
AL_FLANGER_DEFAULT_RATE	Default:0.27f
AL_FLANGER_MIN_DEPTH	Default:0.0f
AL_FLANGER_MAX_DEPTH	Default:1.0f
AL_FLANGER_DEFAULT_DEPTH	Default:1.0f
AL_FLANGER_MIN_FEEDBACK	Default:(-1.0f)
AL_FLANGER_MAX_FEEDBACK	Default:1.0f
AL_FLANGER_DEFAULT_FEEDBACK	Default:(-0.5f)
AL_FLANGER_MIN_DELAY	Default:0.0f
AL_FLANGER_MAX_DELAY	Default:0.004f
AL_FLANGER_DEFAULT_DELAY	Default:0.002f
AL frequency shifter effect parameter	
AL_FREQUENCY_SHIFTER_MIN_FREQUENCY	Default:0.0f
AL_FREQUENCY_SHIFTER_MAX_FREQUENCY	Default:24000.0f
AL_FREQUENCY_SHIFTER_DEFAULT_FREQUENCY	Default:0.0f
AL_FREQUENCY_SHIFTER_MIN_LEFT_DIRECTION	Default:0
AL_FREQUENCY_SHIFTER_MAX_LEFT_DIRECTION	Default:2
AL_FREQUENCY_SHIFTER_DEFAULT_LEFT_DIRECTION	Default:0
AL_FREQUENCY_SHIFTER_MIN_RIGHT_DIRECTION	Default:0
AL_FREQUENCY_SHIFTER_MAX_RIGHT_DIRECTION	Default:2
AL_FREQUENCY_SHIFTER_DEFAULT_RIGHT_DIRECTION	Default:0
AL_FREQUENCY_SHIFTER_DIRECTION_DOWN	Default:0
AL_FREQUENCY_SHIFTER_DIRECTION_UP	Default:1
AL_FREQUENCY_SHIFTER_DIRECTION_OFF	Default:2
AL vocal morphed effect parameter	
AL_VOCAL_MORPHER_MIN_PHONEMEA	Default:0
AL_VOCAL_MORPHER_MAX_PHONEMEA	Default:29
AL_VOCAL_MORPHER_DEFAULT_PHONEMEA	Default:0
AL_VOCAL_MORPHER_MIN_PHONEMEA_COARSE_TUNING	Default:(-24)
AL_VOCAL_MORPHER_MAX_PHONEMEA_COARSE_TUNING	Default:24
AL_VOCAL_MORPHER_DEFAULT_PHONEMEA_COARSE_TUNING	Default:0
AL_VOCAL_MORPHER_MIN_PHONEMEB	Default:0
AL_VOCAL_MORPHER_MAX_PHONEMEB	Default:29
AL_VOCAL_MORPHER_DEFAULT_PHONEMEB	Default:10

AL_VOCAL_MORPHER_PHONEME_A	Default:0
AL_VOCAL_MORPHER_PHONEME_E	Default:1
AL_VOCAL_MORPHER_PHONEME_I	Default:2
AL_VOCAL_MORPHER_PHONEME_O	Default:3
AL_VOCAL_MORPHER_PHONEME_U	Default:4
AL_VOCAL_MORPHER_PHONEME_AA	Default:5
AL_VOCAL_MORPHER_PHONEME_AE	Default:6
AL_VOCAL_MORPHER_PHONEME_AH	Default:7
AL_VOCAL_MORPHER_PHONEME_AO	Default:8
AL_VOCAL_MORPHER_PHONEME_EH	Default:9
AL_VOCAL_MORPHER_PHONEME_ER	Default:10
AL_VOCAL_MORPHER_PHONEME_IH	Default:11
AL_VOCAL_MORPHER_PHONEME_IY	Default:12
AL_VOCAL_MORPHER_PHONEME_UH	Default:13
AL_VOCAL_MORPHER_PHONEME_UW	Default:14
AL_VOCAL_MORPHER_PHONEME_B	Default:15
AL_VOCAL_MORPHER_PHONEME_D	Default:16
AL_VOCAL_MORPHER_PHONEME_F	Default:17
AL_VOCAL_MORPHER_PHONEME_G	Default:18
AL_VOCAL_MORPHER_PHONEME_J	Default:19
AL_VOCAL_MORPHER_PHONEME_K	Default:20
AL_VOCAL_MORPHER_PHONEME_L	Default:21
AL_VOCAL_MORPHER_PHONEME_M	Default:22
AL_VOCAL_MORPHER_PHONEME_N	Default:23
AL_VOCAL_MORPHER_PHONEME_P	Default:24
AL_VOCAL_MORPHER_PHONEME_R	Default:25
AL_VOCAL_MORPHER_PHONEME_S	Default:26
AL_VOCAL_MORPHER_PHONEME_T	Default:27
AL_VOCAL_MORPHER_PHONEME_V	Default:28
AL_VOCAL_MORPHER_PHONEME_Z	Default:29
AL_VOCAL_MORPHER_MIN_PHONEMEB_COARSE_TUNING	Default:(-24)
AL_VOCAL_MORPHER_MAX_PHONEMEB_COARSE_TUNING	Default:24
AL_VOCAL_MORPHER_DEFAULT_PHONEMEB_COARSE_TUNING	Default:0
AL_VOCAL_MORPHER_MIN_WAVEFORM	Default:0
AL_VOCAL_MORPHER_MAX_WAVEFORM	Default:2
AL_VOCAL_MORPHER_DEFAULT_WAVEFORM	Default:0
AL_VOCAL_MORPHER_WAVEFORM_SINUSOID	Default:0
AL_VOCAL_MORPHER_WAVEFORM_TRIANGLE	Default:1
AL_VOCAL_MORPHER_WAVEFORM_SAWTOOTH	Default:2
AL_VOCAL_MORPHER_MIN_RATE	Default:0.0f
AL_VOCAL_MORPHER_MAX_RATE	Default:10.0f
AL_VOCAL_MORPHER_DEFAULT_RATE	Default:1.41f
AL pitch shifter effect parameter	

AL_PITCH_SHIFTER_MIN_COARSE_TUNE	Default:(-12)
AL_PITCH_SHIFTER_MAX_COARSE_TUNE	Default:12
AL_PITCH_SHIFTER_DEFAULT_COARSE_TUNE	Default:12
AL_PITCH_SHIFTER_MIN_FINE_TUNE	Default:(-50)
AL_PITCH_SHIFTER_MAX_FINE_TUNE	Default:50
AL_PITCH_SHIFTER_DEFAULT_FINE_TUNE	Default:0
AL ring modulator effect parameter	
AL_RING_MODULATOR_MIN_FREQUENCY	Default:0.0f
AL_RING_MODULATOR_MAX_FREQUENCY	Default:8000.0f
AL_RING_MODULATOR_DEFAULT_FREQUENCY	Default:440.0f
AL_RING_MODULATOR_MIN_HIGHPASS_CUTOFF	Default:0.0f
AL_RING_MODULATOR_MAX_HIGHPASS_CUTOFF	Default:24000.0f
AL_RING_MODULATOR_DEFAULT_HIGHPASS_CUTOFF	Default:800.0f
AL_RING_MODULATOR_MIN_WAVEFORM	Default:0
AL_RING_MODULATOR_MAX_WAVEFORM	Default:2
AL_RING_MODULATOR_DEFAULT_WAVEFORM	Default:0
AL_RING_MODULATOR_SINUSOID	Default:0
AL_RING_MODULATOR_SAWTOOTH	Default:1
AL_RING_MODULATOR_SQUARE	Default:2
AL Autowah effect parameter	
AL_AUTOWAH_MIN_ATTACK_TIME	Default:0.0001f
AL_AUTOWAH_MAX_ATTACK_TIME	Default:1.0f
AL_AUTOWAH_DEFAULT_ATTACK_TIME	Default:0.06f
AL_AUTOWAH_MIN_RELEASE_TIME	Default:0.0001f
AL_AUTOWAH_MAX_RELEASE_TIME	Default:1.0f
AL_AUTOWAH_DEFAULT_RELEASE_TIME	Default:0.06f
AL_AUTOWAH_MIN_RESONANCE	Default:2.0f
AL_AUTOWAH_MAX_RESONANCE	Default:1000.0f
AL_AUTOWAH_DEFAULT_RESONANCE	Default:1000.0f
AL_AUTOWAH_MIN_PEAK_GAIN	Default:0.00003f
AL_AUTOWAH_MAX_PEAK_GAIN	Default:31621.0f
AL_AUTOWAH_DEFAULT_PEAK_GAIN	Default:11.22f
AL compressor effect parameter	
AL_COMPRESSOR_MIN_ONOFF	Default:0
AL_COMPRESSOR_MAX_ONOFF	Default:1
AL_COMPRESSOR_DEFAULT_ONOFF	Default:1
AL equalizer effect parameter	

AL_EQUALIZER_MIN_LOW_GAIN	Default:0.126f
AL_EQUALIZER_MAX_LOW_GAIN	Default:7.943f
AL_EQUALIZER_DEFAULT_LOW_GAIN	Default:1.0f
AL_EQUALIZER_MIN_LOW_CUTOFF	Default:50.0f
AL_EQUALIZER_MAX_LOW_CUTOFF	Default:800.0f
AL_EQUALIZER_DEFAULT_LOW_CUTOFF	Default:200.0f
AL_EQUALIZER_MIN_MID1_GAIN	Default:0.126f
AL_EQUALIZER_MAX_MID1_GAIN	Default:7.943f
AL_EQUALIZER_DEFAULT_MID1_GAIN	Default:1.0f
AL_EQUALIZER_MIN_MID1_CENTER	Default:200.0f
AL_EQUALIZER_MAX_MID1_CENTER	Default:3000.0f
AL_EQUALIZER_DEFAULT_MID1_CENTER	Default:500.0f
AL_EQUALIZER_MIN_MID1_WIDTH	Default:0.01f
AL_EQUALIZER_MAX_MID1_WIDTH	Default:1.0f
AL_EQUALIZER_DEFAULT_MID1_WIDTH	Default:1.0f
AL_EQUALIZER_MIN_MID2_GAIN	Default:0.126f
AL_EQUALIZER_MAX_MID2_GAIN	Default:7.943f
AL_EQUALIZER_DEFAULT_MID2_GAIN	Default:1.0f
AL_EQUALIZER_MIN_MID2_CENTER	Default:1000.0f
AL_EQUALIZER_MAX_MID2_CENTER	Default:8000.0f
AL_EQUALIZER_DEFAULT_MID2_CENTER	Default:3000.0f
AL_EQUALIZER_MIN_MID2_WIDTH	Default:0.01f
AL_EQUALIZER_MAX_MID2_WIDTH	Default:1.0f
AL_EQUALIZER_DEFAULT_MID2_WIDTH	Default:1.0f
AL_EQUALIZER_MIN_HIGH_GAIN	Default:0.126f
AL_EQUALIZER_MAX_HIGH_GAIN	Default:7.943f
AL_EQUALIZER_DEFAULT_HIGH_GAIN	Default:1.0f
AL_EQUALIZER_MIN_HIGH_CUTOFF	Default:4000.0f
AL_EQUALIZER_MAX_HIGH_CUTOFF	Default:16000.0f
AL_EQUALIZER_DEFAULT_HIGH_CUTOFF	Default:6000.0f
Source parameter value definitions	
AL_MIN_AIR_ABSORPTION_FACTOR	Default:0.0f
AL_MAX_AIR_ABSORPTION_FACTOR	Default:10.0f
AL_DEFAULT_AIR_ABSORPTION_FACTOR	Default:0.0f
AL_MIN_ROOM_ROLLOFF_FACTOR	Default:0.0f
AL_MAX_ROOM_ROLLOFF_FACTOR	Default:10.0f

AL_DEFAULT_ROOM_ROLLOFF_FACTOR	Default:0.0f
AL_MIN_CONE_OUTER_GAINHF	Default:0.0f
AL_MAX_CONE_OUTER_GAINHF	Default:1.0f
AL_DEFAULT_CONE_OUTER_GAINHF	Default:1.0f
AL_MIN_DIRECT_FILTER_GAINHF_AUTO	Default:AL_FALSE
AL_MAX_DIRECT_FILTER_GAINHF_AUTO	Default:AL_TRUE
AL_DEFAULT_DIRECT_FILTER_GAINHF_AUTO	Default:AL_TRUE
AL_MIN_AUXILIARY_SEND_FILTER_GAIN_AUTO	Default:AL_FALSE
AL_MAX_AUXILIARY_SEND_FILTER_GAIN_AUTO	Default:AL_TRUE
AL_DEFAULT_AUXILIARY_SEND_FILTER_GAIN_AUTO	Default:AL_TRUE
AL_MIN_AUXILIARY_SEND_FILTER_GAINHF_AUTO	Default:AL_FALSE
AL_MAX_AUXILIARY_SEND_FILTER_GAINHF_AUTO	Default:AL_TRUE
AL_DEFAULT_AUXILIARY_SEND_FILTER_GAINHF_AUTO	Default:AL_TRUE
Listener parameter value definitions	
AL_MIN_METERS_PER_UNIT	Default:FLT_MIN
AL_MAX_METERS_PER_UNIT	Default:FLT_MAX
AL_DEFAULT_METERS_PER_UNIT	Default:1.0f

ΠΙΝΑΚΑΣ 4.1: ALUT Functions list	
Initialization / Exit Functions	
alutInit	Αρχικοποίηση τιμών για την βιβλιοθήκη ALUT και δημιουργία ενός τρέχοντος OpenAL context για το default device .
alutInitWithoutContext	Αρχικοποίηση τιμών για την βιβλιοθήκη ALUT. Δεν δημιουργεί κανένα τρέχον OpenAL context για το default device. Έτσι αυτό πρέπει να γίνει μέσω των συνηθισμένων ALC κλήσεων
alutExit	Κλείσιμο της βιβλιοθήκης ALUT. Αυτό κλείνει οποιαδήποτε OpenAL device / context έχουν δημιουργηθεί με την alutIni (αλλά όχι αυτά που η εφαρμογή δημιούργησε με την χρησιμοποίηση της ALC).
Error Functions	
alutGetError	Ανακτά και εκκαθαρίζει την επικρατούσα κατάσταση λάθους ALUT.
alutGetErrorString	Επιστρέφει ένα μήνυμα σφάλματος, έχοντας ως δεδομένο ένα κώδικα σφάλματος (alutGetError)
Sound Sample File Loading Functions	
alutCreateBufferFromFile	Φορτώνει ένα αρχείο ήχου σε έναν OpenAL buffer (προσωρινή αποθήκευση στοιχείων - ενδιάμεση μνήμη)
alutCreateBufferFromFileImage	Φορτώνει in-memory αρχείο ήχου σε έναν OpenAL buffer
alutCreateBufferHelloWorld	Δημιουργεί ένα buffer με ένα ήχου 'Hello, world!'
alutCreateBufferWaveform	Δημιουργεί έναν buffer με έναν πίνακα κυματομορφών
alutLoadMemoryFromFile	Φορτώνει ένα αρχείο ήχου μέσα στο OpenAL-like data
alutLoadMemoryFromFileImage	Μετατρέπει τα in-memory αρχείο ήχου σε OpenAL- like data.
alutLoadMemoryHelloWorld	Φορτώνει το 'Hello, world!' ήχο μέσα στο OpenAL - like data
alutLoadMemoryWaveform	Φορτώνει των πίνακα κυματομορφών μέσα στο OpenAL - like data
alutGetMIMETypes	Παρέχει μια λίστα με τους παρεχόμενους audio MIME types.

WAV loaders σε απόσυρση: Για την προς τα πίσω-συμβατότητα με τις εκδόσεις ALUT 0.x.x, η ALUT:	
alutLoadWAVFile	Φορτώνει ένα wav ήχο σε έναν OpenAL buffer
alutLoadWAVMemory	Φορτώνει in-memory wav ήχο σε έναν OpenAL buffer.
alutUnloadWAV	Ξεφορτώνει ένα wav αρχείο ήχου από τον buffer.
Version Checking Functions	
alutGetMajorVersion	Επιστρέφει τον αριθμό έκδοσης ALUT major.
alutGetMinorVersion	Επιστρέφει τον αριθμό έκδοσης ALUT Minor.
Sleeping Functions	
alutSleep	Σταματάει την εφαρμογή για έναν δεδομένο αριθμό δευτερολέπτων.

ΠΙΝΑΚΑΣ 4.2 ALUT Define Values

Error Values	
ALUT_ERROR_NO_ERROR	Κανένα ALUT λάθος δεν βρέθηκε
ALUT_ERROR_OUT_OF_MEMORY	Η μνήμη δεν επαρκεί.
ALUT_ERROR_INVALID_ENUM	Δόθηκε μια άκυρη τιμή enum. Άκυρη παράμετρος (bad enum).
ALUT_ERROR_INVALID_VALUE	Δόθηκε μια άκυρη τιμή .
ALUT_ERROR_INVALID_OPERATION	Η λειτουργία είναι άκυρη στην επικρατούσα κατάσταση ALUT.
ALUT_ERROR_NO_CURRENT_CONTEXT	Δεν υπάρχει κανένα τρέχον AL context.
ALUT_ERROR_AL_ERROR_ON_ENTRY	Υπήρξε ήδη ένα λάθος AL κατά την εισαγωγή σε μια λειτουργία ALUT.
ALUT_ERROR_ALC_ERROR_ON_ENTRY	Υπήρξε ήδη ένα λάθος ALC κατά την εισαγωγή σε μια λειτουργία ALUT.
ALUT_ERROR_OPEN_DEVICE	Υπήρξε ένα λάθος κατά το άνοιγμα ενός ALC device
ALUT_ERROR_CLOSE_DEVICE	Υπήρξε ένα λάθος κατά το κλείσιμο ενός ALC device
ALUT_ERROR_CREATE_CONTEXT	Υπήρξε ένα λάθος κατά την δημιουργία ενός ALC context .
ALUT_ERROR_MAKE_CONTEXT_CURRENT	Δεν μπορεί να τεθεί ένα context ως το τρέχον ALC context .
ALUT_ERROR_DESTROY_CONTEXT	Υπήρξε ένα λάθος κατά την καταστροφή του ALC context
ALUT_ERROR_GEN_BUFFERS	Υπήρξε ένα λάθος στην δημιουργία ενός AL buffer

ALUT_ERROR_BUFFER_DATA	Υπήρξε ένα λάθος κατά την διάρκεια εισαγωγής στοιχείων στον buffer AL
ALUT_ERROR_IO_ERROR	I/O error. Συμβουλευτείτε το errno για περισσότερες λεπτομέρειες
ALUT_ERROR_UNSUPPORTED_FILE_TYPE	Μη υποστηρίξιμος τύπος αρχείου.
ALUT_ERROR_UNSUPPORTED_FILE_SUBTYPE	Μη υποστηρίξιμος επιλογή εντός μιας υποκατηγορίας ενός ήδη χρησιμοποιημένου τύπου αρχείου
ALUT_ERROR_CORRUPT_OR_TRUNCATED_DATA	Το αρχείο ήχου έχει αλλοιωθεί ή έχει περικοπεί
Waveforms (Χρήση με alutCreateBufferWaveform)	
ALUT_WAVEFORM_SINE	Επιτρεπόμενος - διαθέσιμος τύπος κυματομορφής
ALUT_WAVEFORM_SQUARE	Επιτρεπόμενος - διαθέσιμος τύπος κυματομορφής
ALUT_WAVEFORM_SAWTOOTH	Επιτρεπόμενος - διαθέσιμος τύπος κυματομορφής
ALUT_WAVEFORM_WHITENOISE	Επιτρεπόμενος - διαθέσιμος τύπος κυματομορφής
ALUT_WAVEFORM_IMPULSE	Επιτρεπόμενος - διαθέσιμος τύπος κυματομορφής
Loader Types(Χρήση με alutGetMIMETypes)	
ALUT_LOADER_BUFFER	Για τους loaders που επιστρέφουν τα αρχεία ήχου σε έναν OpenAL buffer, π.χ. alutCreateBufferFromFile και alutCreateBufferFromFileImage
ALUT_LOADER_MEMORY	Για τους loaders που επιστρέφουν τα αρχεία ήχου σε μια πρόσφατα διατιθέμενη περιοχή μνήμης, π.χ. alutLoadMemoryFromFile και alutLoadMemoryFromFileImage

ΠΑΡΑΡΤΗΜΑ 2 Development Platform- Compiler & IDE, Εγκατάσταση ενός API

Στο παράρτημα αυτό περιγράφονται, βήμα προς βήμα, όλες οι απαραίτητες ενέργειες, ώστε να εγκατασταθούν σωστά και να είναι έτοιμα για μεταγλώττιση δυο από τα πιο γνωστά περιβάλλοντα προγραμματισμού Eclipse(C\C++) και Visual studio(C++). Επίσης δίνονται οι απαραίτητες πληροφορίες για πως μπορεί να ενσωματωθεί ένα API σε ένα *IDE* (*Integrated Development Environment –ολοκληρωμένο περιβάλλον ανάπτυξης*), έχοντας ως παράδειγμα το API OpenAL, μιας και αποτελεί το κύριο αντικείμενο της συγκεκριμένης πτυχιακής. Η πλειοψηφία των API μπορούν να ενσωματωθούν σε ένα IDE με τον ίδιο ακριβώς τρόπο. [62]

Όλες οι ενέργειες που περιγράφονται στο παρόν παράρτημα αναφέρονται στο λειτουργικό σύστημα των windows.

1. Εγκατάσταση στο περιβάλλον του Eclipse

Το Eclipse αναπτύσσεται από μια μεγάλη κοινότητα προγραμματιστών. Αποτελείται από ένα μεγάλο αριθμό εργαλείων ανάπτυξης, διαχείρισης, frameworks κ.α.., που αναπτύσσονται ανεξάρτητα από υποομάδες της κοινότητας. Είναι ένα Open Source IDE που παρέχεται εντελώς δωρεάν. [63]

Αυτή τη στιγμή διανέμονται ξεχωριστά IDE για προγραμματισμό σε πολλές γλώσσες, ενώ υπάρχει και δυνατότητα χρήσης ενός κεντρικού IDE (Eclipse Classic), που να παρέχει όλα τα εργαλεία ανάπτυξης και σε όλες της γλώσσες προγραμματισμού, που παρέχονται από την κοινότητα του Eclipse. [63]

Εικόνα 1 Το λογότυπο του Eclipse

Αυτή τη στιγμή υπάρχουν δύο επιλογές για την εγκατάσταση του eclipse σε C\C++:

Η πρώτη επιλογή είναι η εγκατάσταση του πακέτου Eclipse IDE for C/C++ Developers που έχει προεγκατεστημένο το C/C++ Development. [63]

Η δεύτερη επιλογή είναι η εγκατάσταση του πακέτου Eclipse Classic, όπου μέσω update μπορούμε να εγκαταστήσουμε το C/C++ Development Tooling (CDT). [63]

Και τα δύο παρέχονται στη διεύθυνση: http://www.eclipse.org/downloads/ (Πρόσβαση: 10 Οκτωβρίου 2006)

Μετά την επιλογή του πακέτου είναι επίσης απαραίτητο να εγκατασταθεί το Java 5 runtime environment (JRE) από την διεύθυνση: http://java.sun.com/. Η Java είναι απαραίτητη ακόμα και αν δεν θέλουμε να εργαστούμε σε περιβάλλον java, μιας και το Eclipse είναι εξολοκλήρου γραμμένο σε java. [63]

Στην περίπτωση που επιλέξουμε το Eclipse Classic πρέπει να εγκαταστήσουμε το C/C++ Development Tooling (CDT), το οποίο είναι ένα εργαλείο που επιτρέπει την ανάπτυξη κώδικα σε C/C ++ χρησιμοποιώντας το Eclipse. [63,64]

Βήματα εγκατάστασης του CDT: *Help -> Software Updates -> Find and Install*. Επιλέγουμε "*Search for New Features to Install*" και κάνουμε click και στα δύο "*Callisto Discovery Site*" και "*The Eclipse Project Updates*". Επιλέγουμε "*Update Site Mirror*" και εγκαταθιστούμε οποιοδήποτε update για το Eclipse και έπειτα εγκαθιστούμε "C and C++ Development" (CDT) από την περιοχή Callisto Discovery site. [63,64]

Παράλληλα, υπάρχουν και οδηγίες που μπορεί κανείς να ακολουθήσει, με βάση τα βήματα από το video που βρίσκεται στη διεύθυνση: *http://mirror.yoxos-eclipse-distribution.de/eclipse.org/technology/phoenix/demos/install-cdt/callisto-cdt.html*
(Πρόσβαση : 10 Οκτωβρίου 2006)

Το Eclipse με CDT Tooling και το Eclipse IDE for C/C++ Developers δεν έχουν προεγκατεστημένο έναν μεταγλωττιστή της C/C++ και έτσι χρειάζεται να εγκατασταθεί ένας εξαρχής. Το GCC είναι μία από τις προτεινόμενες επιλογές. Αυτό σημαίνει ότι πρέπει να εγκαταστήσουμε το GCC και να το καταστήσουμε διαθέσιμο στους πόρους του συστήματός μας (system path). [63,64]

Εικόνα 2 Το λογότυπο του GCC

Για την εγκατάσταση του GCC έχουμε τρεις επιλογές :

Η πρώτη επιλογή είναι το Cygwin (GCC version 3.4.4), που μεταφράζει απλά όλες τις κλήσεις του συστήματος Unix σε κλήσεις συστήματος windows, επιτρέποντας έτσι σε όλα τα προγράμματα Unix να τρέχουν στα windows με τις ελάχιστες απαιτήσεις. [64,65]

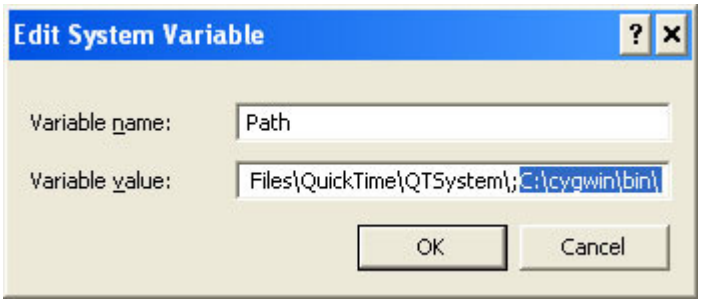
Η δεύτερη επιλογή είναι το MinGW (GCC version 3.4.5). Το MinGW είναι μια συλλογή από Linux development tools (όπως compilers και header files), που περιλαμβάνει και GCC για τα windows. [66,67,68]

Η τρίτη επιλογή είναι GccWinBinaries (gcc version 4.1.2).Το GccWinBinaries GCC περιέχει έναν γρήγορο installer του GCC 4.1.2 για τα Windows. Είναι μια ανεπίσημη έκδοση, που δεν προωθείται από το πρόγραμμα MinGW. Οι υπεύθυνοι για την ανάπτυξη του MinGW θεωρούν ότι το GCC 4.x δεν είναι έτοιμο για μια επίσημη έκδοση. [69]

1.1 Eclipse CDT & Cygwin (GCC Version 3.4.4)

Το Cygwin βρίσκεται στη διεύθυνση: *http://www.cygwin.com/setup.exe* (Πρόσβαση:10 Οκτωβρίου 2006). Εκτελούμε το πρόγραμμα εγκατάστασης Cygwin και κατά την εγκατάσταση επιλέγουμε τις επιλογές GCC από την κατηγορία development category ή μπορούμε να εγκαταστήσουμε και αλλά development tools. [64]

Μόλις ολοκληρωθεί η εγκατάσταση, προσθέτουμε το Cygwin στον κατάλογο πόρων του συστήματος μας. Ουσιαστικά πρέπει να δημιουργήσουμε ένα system path για το Cygwin, ώστε να μπορεί να το βρει οποιαδήποτε εφαρμογή μπορεί να το χρησιμοποιήσει (*Start->Settings->Control panel->System->Advanced->Environment variables->Path*). Click Edit και προσθέστε *;c:\cygwin\bin* [64]



Εικόνα 3 Προσθήκη του Cygwin στα system path του υπολογιστή

Σε περίπτωση που έχουμε ήδη ανοικτό το eclipse πρέπει να γίνει επανεκκίνηση, ώστε να ενημερωθεί το σύστημά μας με τους νέους πόρους συστήματος. Μπορούμε να ελέγξουμε αν έχουμε εγκαταστήσει σωστά το Cygwin από το *Start Menu -> Run,* πληκτρολογώντας την εντολή *gcc –v* και πατώντας enter. Αν έχει γίνει σωστά η εγκατάσταση θα πρέπει να πάρουμε κάποιες πληροφορίες έκδοσης (Version 3.4.4). Αν το αποτέλεσμα είναι *'not recognized as an internal or external command'*, πρέπει να ελέγξουμε αν έχουμε δηλώσει την σωστή διαδρομή στους πόρους συστήματος. [67]

1.2 Eclipse CDT & MinGW (GCC Version 3.4.5)

Περίπου την ίδια διαδικασία του Cygwin, ακολουθούμε και για το MinGW. Το MinGW Toolbox βρίσκεται στη διεύθυνση:

http://kent.dl.sourceforge.net/sourceforge/ogre/MinGW_Toolbox_Setup_wr1.exe

(Πρόσβαση: 10 Οκτωβρίου 2006)

Εκτελούμε το πρόγραμμα MinGW Toolbox Setup χωρίς καμία απολύτως τροποποίηση. Μόλις ολοκληρωθεί η εγκατάσταση, προσθέτουμε το MinGW στον κατάλογο πόρων του συστήματος μας. (*Start->Settings->Control panel->System-> Advanced -> Environment variables -> Path)*.Click Edit και προσθέτουμε *;c:\mingw\bin* [64,68]

Επίσης πρέπει να εγκαταστήσουμε το MSYS για να υπάρχει πρόσβαση στις εντολές Linux-style. Το MSYS είναι διαθέσιμο στην ακόλουθη διεύθυνση: http://prdownloads.sf.net/mingw/MSYS-1.0.10.exe?download (Πρόσβαση: 10 Οκτωβρίου 2006). Κατά την εγκατάσταση θα μας ζητηθεί να υποδείξουμε που είναι το MinGW όπου δηλώνουμε *c:\mingw*. Μετά το τέλος της εγκατάστασης πρέπει να δηλώσουμε και το MSYS στους πόρους συστήματος όπου προσθέτουμε *;c:\msys\bin* [68]

Σε περίπτωση που έχουμε ήδη ανοικτό το eclipse πρέπει να γίνει επανεκκίνηση, ώστε να ενημερωθεί το σύστημά μας με τους νέους πόρους συστήματος. Μπορούμε να ελέγξουμε αν έχουμε εγκαταστήσει σωστά το MinGW από το *Start Menu -> Run* πληκτρολογώντας την εντολή *gcc –v* και πατώντας enter. Αν έχει γίνει σωστά η εγκατάσταση θα πρέπει να πάρουμε κάποιες πληροφορίες έκδοσης (Version 3.4.5). Αν το αποτέλεσμα είναι *'not recognized as an internal or external command'*, πρέπει να ελέγξουμε αν έχουμε δηλώσει την σωστή διαδρομή στους πόρους συστήματος. [67]

1.3 Eclipse CDT & GccWinBinaries (GCC Version 4.1.2)

Πρέπει να επανασημειωθεί ότι το GccWinBinaries είναι μια ανεπίσημη έκδοση, που δεν προωθείται από το πρόγραμμα MinGW. Οι υπεύθυνοι για την ανάπτυξη του MinGW θεωρούν ότι το GCC 4.x δεν είναι έτοιμο για μια επίσημη έκδοση. Επομένως στην περίπτωση που επιλέξουμε το συγκεκριμένο πακέτο πρέπει να είμαστε ιδιαίτερα προσεκτικοί, μιας και τυχόν προβλήματα που μπορεί να παρουσιαστούν μπορεί να οφείλονται στο γεγονός αυτό. [69]

Το GccWinBinaries είναι διαθέσιμο στη διεύθυνση:

http://www.develer.com/~rasky/gcc-4.1.2-mingw-setup.exe

(Πρόσβαση: 10 Οκτωβρίου 2006)

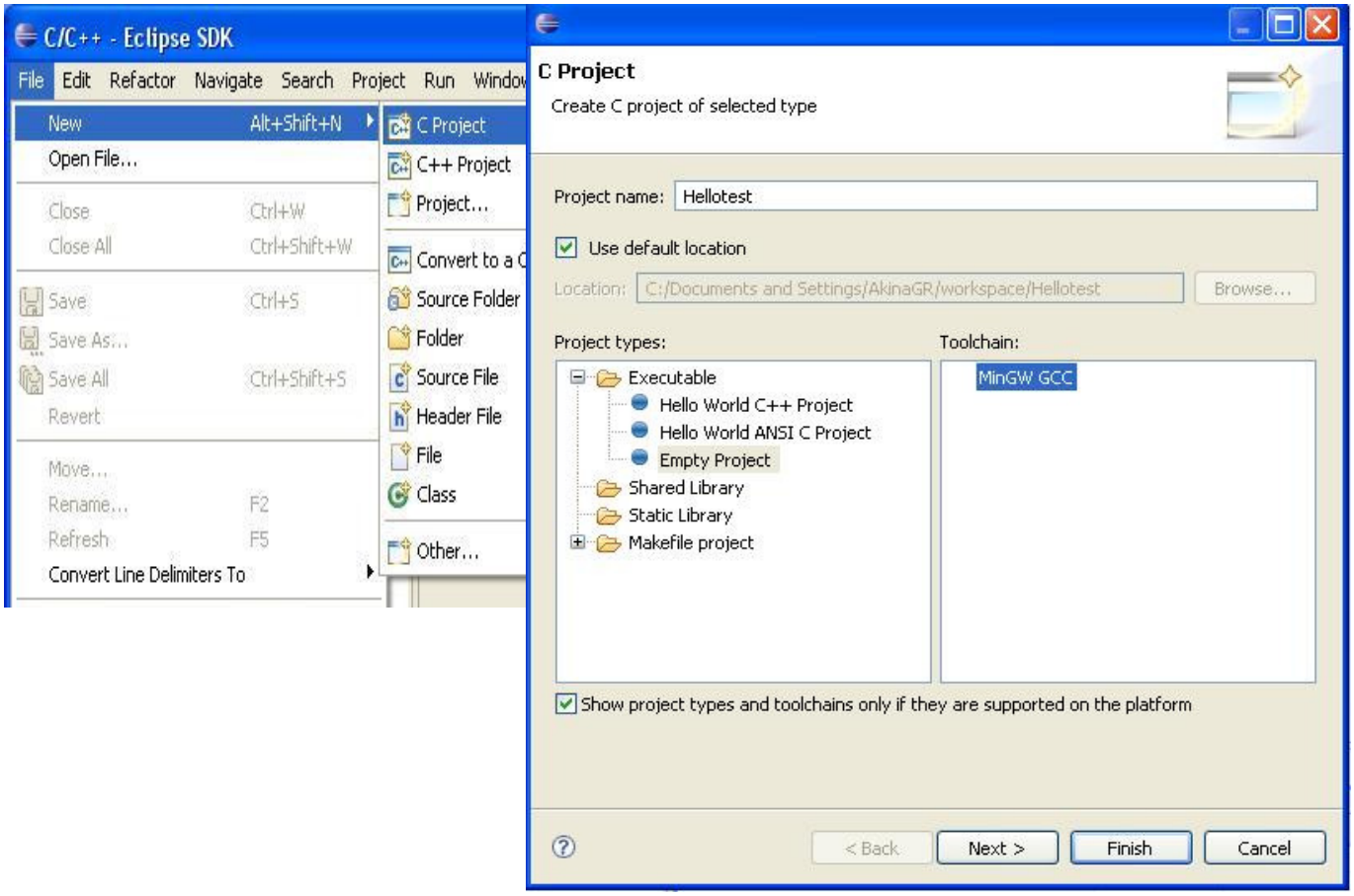
Εκτελούμε το πρόγραμμα εγκατάστασης GccWinBinaries και όταν εμφανιστεί η καρτέλα επιλογών επιλέγουμε: *"Add GCC to your system is PATH"*.

Μετά το τέλος της εγκατάστασης πρέπει να ακολουθήσουμε ακριβώς την ίδια διαδικασία που αναφέρουμε στο υποκεφάλαιο 1.2 Eclipse CDT & MinGW (GCC Version 3.4.5). Δηλαδή εγκατάσταση του MSYS, δήλωση στους πόρους συστήματος του MSYS και εξέταση για την σωστή εγκατάσταση μέσω της εντολής *gcc –v,* όπου εδώ θα πρέπει να εμφανιστεί η πληροφορία έκδοσης (GCC Version 4.1.2.).

1.4 Δημιουργία ένας νέου C/C++ project

Για το κεφαλαίο αυτό επιλέξαμε το IDE Eclipse Classic, επειδή η διαδικασία είναι λίγο πιο περίπλοκη. Σε γενικές γραμμές και στα δύο IDE Eclipse Classic και Eclipse IDE for C/C++ Developers οι διαδικασίες που περιγράφουμε είναι πανομοιότυπες.

Αρχικά πρέπει να δημιουργήσουμε ένα νέο Project στο *Eclipse File -> New -> C Project ->Managed make C project -> next* και να επιλέξουμε ένα νέο όνομα για το Project, *next-> finish.* Επιλέγουμε ένα project name, την έκδοση του GCC που θα χρησιμοποιήσουμε (π.χ. MinGW GCC), τον τύπο του project που θα δημιουργήσουμε (π.χ Empty project) και επιλέγουμε *finish,* όπως παρουσιάζεται στην εικόνα 4. [64,68]

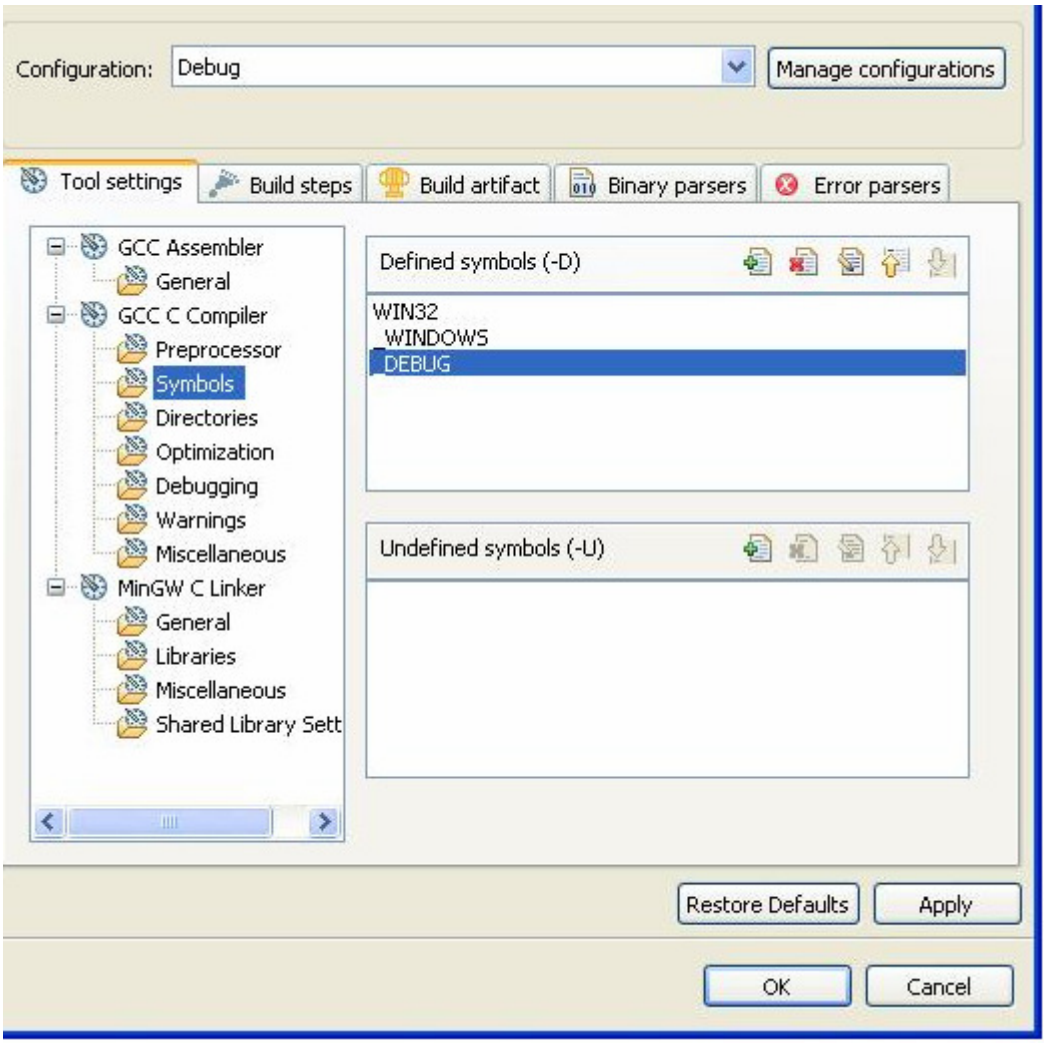


Εικόνα 4 Δημιουργία ενός νέου project.

Στην συνέχεια πρέπει να κάνουμε μια τροποποίηση στις ιδιότητες προγράμματος, ανεξαρτήτως του πακέτου GCC που έχουμε επιλέξει. Δεξί click στο όνομα του Project που δημιουργήσαμε και επιλέγουμε *Properties* ή Alt +Enter. Επιλέγουμε *C/C++ Build -> Settings* και στην ενότητα *GCC C Compiler -> Symbols* εισάγουμε τα παρακάτω: *Defined Symbols (-D)* για *Debug* και *Release* αντίστοιχα, όπως φαίνονται από τον πίνακα που ακολουθεί. Αφού συμπληρώσουμε τα Defined Symbols επιλέγουμε το ok, ώστε να αποθηκευτούν οι ρυθμίσεις μας. [68]

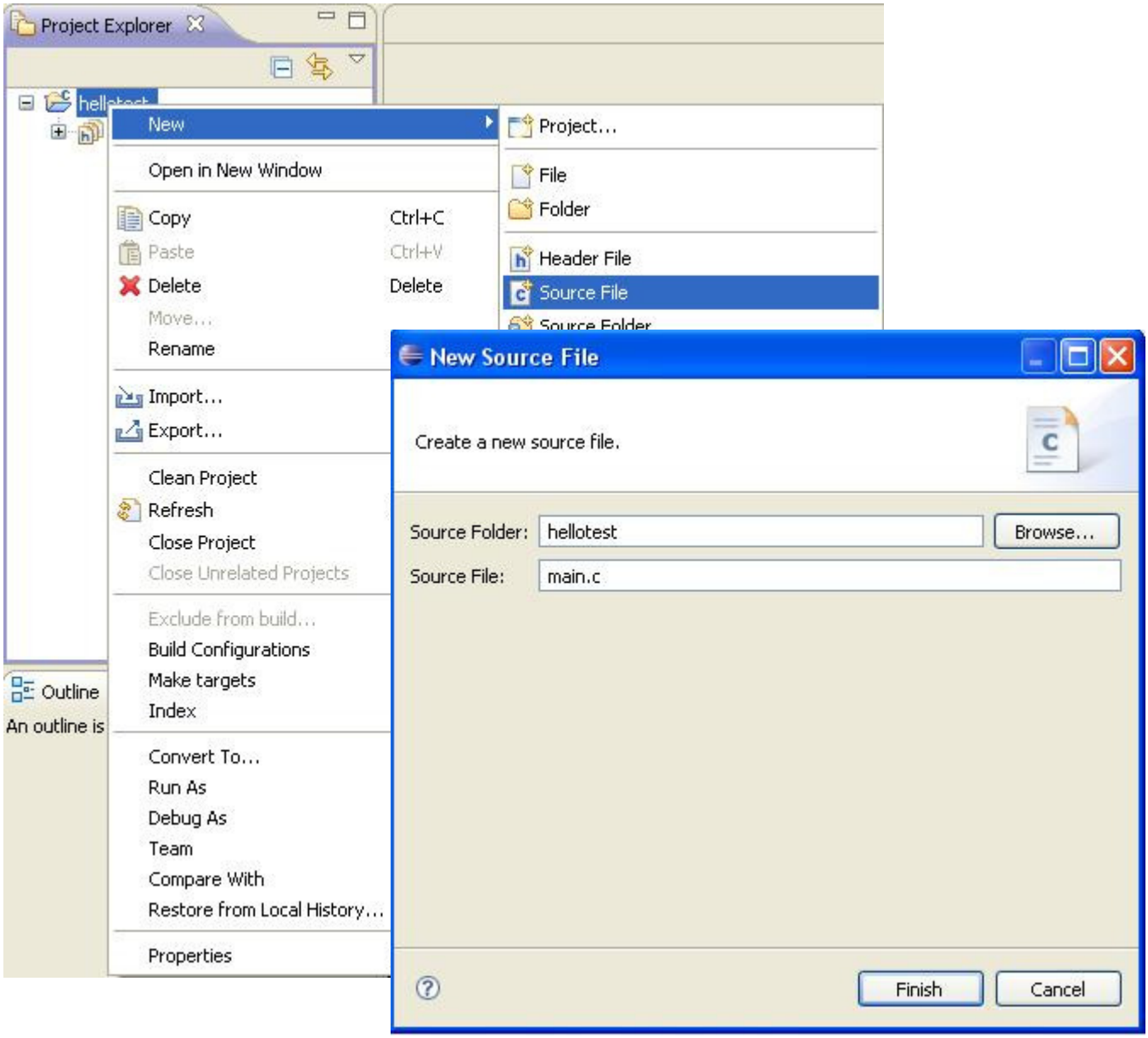
DEBUG	RELEASE
WIN32 _WINDOWS _DEBUG	WIN32 _WINDOWS NDEBUG

Πίνακας 1 GCC C Compiler Defined Symbols



Εικόνα 5 GCC C Compiler Defined Symbols

Επιλέγουμε δεξί click στο όνομα του Project, *New->Other.. -> C->Source file* και δημιουργούμε ένα αρχείο κώδικα με το όνομα που επιθυμούμε (Εικόνα 6).



Εικόνα 6 Δημιουργία ενός νέου *Source file*

Στην συνέχεια πληκτρολογούμε ένα απλό παράδειγμα κώδικα, σαν αυτό που φαίνεται παρακάτω:

```
#include <stdio.h>
int main(void) {
printf("Hello, world C!\n");
return 0;
}
```

Αποθηκεύουμε το αρχείο (ctrl+s) και το Eclipse θα κάνει αυτόματα build το αρχείο. Στην console του Eclipse πρέπει να εμφανιστεί ένα μήνυμα σαν αυτό *«Build complete for project name. Time consumed: 625 ms. »*

2. Εγκατάσταση στο περιβάλλον της Microsoft Visual Express Edition

Καταρχήν πρέπει να αναφέρουμε ότι οι εκδόσεις Express του Visual studio διανέμονται δωρεάν, με μοναδική επιβάρυνση ένα user register, που πρέπει να κάνει ο κάθε χρήστης. Η προηγούμενη Express έκδοση του Visual studio για C++ απαιτούσε 5 βήματα κατά την εγκατάσταση, ώστε να είναι δυνατή η λειτουργία του μεταγλωττιστή. Πλέον η Microsoft έχει αποσύρει τις σελίδες που αναθέτανε τις οδηγίες εγκατάστασης για το Visual studio Express 2005 και διανέμει σε beta 2 το Visual Studio 2008 Express.[70,71]

Εικόνα 7 Το λογότυπο του Visual studio

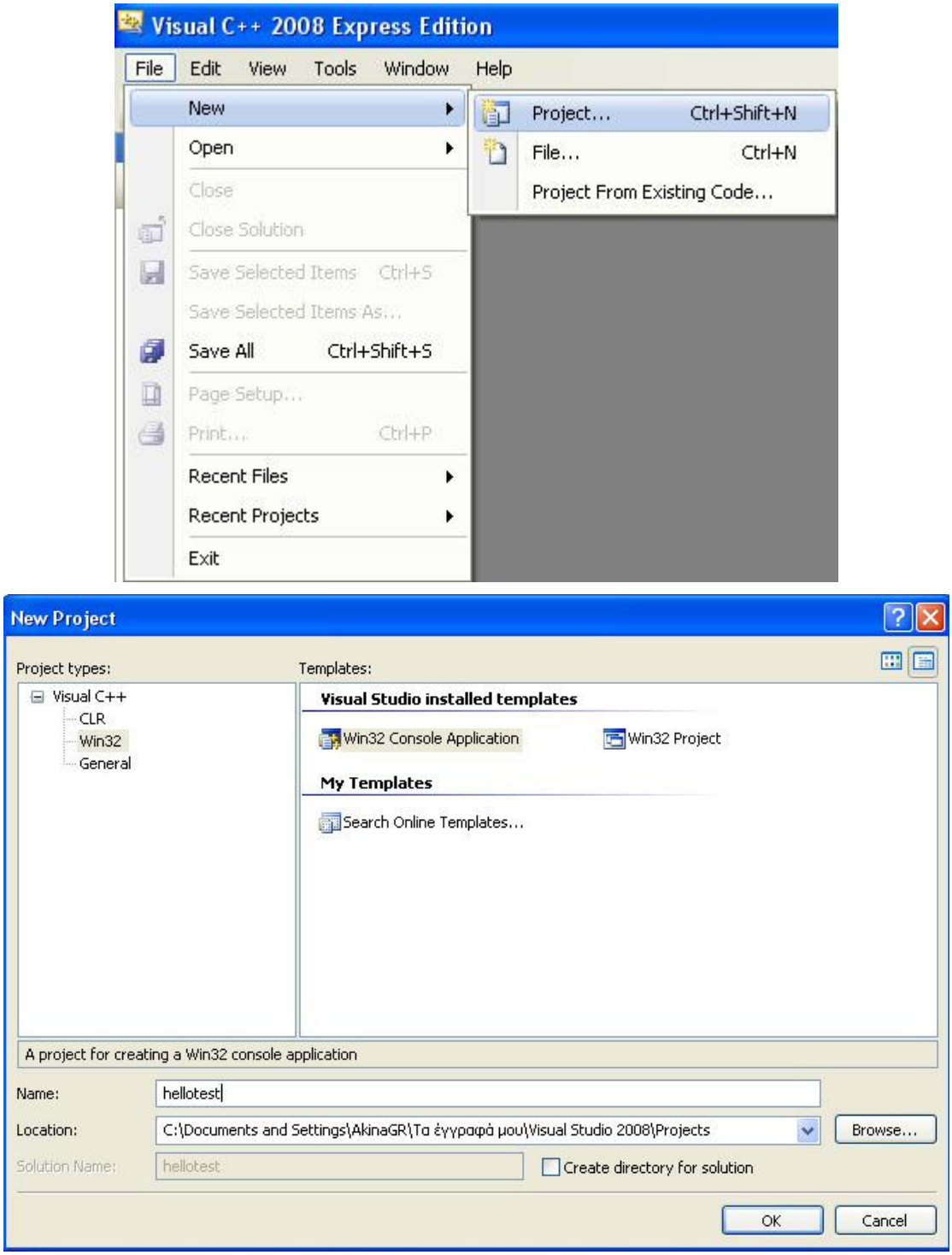
Σε περίπτωση που επιθυμούμε τη χρήση της σταθερής έκδοσης Visual studio Express 2005 τα 5 βήματα που χρειάζονται μπορούν να βρεθούν σε αυτή τη διεύθυνση: http://www.codeproject.com/useritems/FreeVS2005Win32.asp (Πρόσβαση: 10 Οκτωβρίου 2006).

Στις συγκεκριμένες οδηγίες χρήσης θα χρησιμοποιήσουμε το Visual Studio 2008 Express. Πριν την εγκατάσταση οποιασδήποτε έκδοσης του Visual studio πρέπει να ελέγξουμε αν είναι ήδη εγκατεστημένο το NET Framework και σε περίπτωση που δεν είναι εγκατεστημένο να το εγκαταστήσουμε. Καλό θα ήταν, επίσης, να εγκαταστήσουμε και το Microsoft Platform SDK, ανεξάρτητα από το αν χρειάζεται για την έκδοση Visual studio που θα χρησιμοποιήσουμε, μιας και περιέχει ένα μεγάλο αριθμό βιβλιοθηκών και ενσωματωμένα API, που πολλές φορές είναι χρήσιμα ακόμα και από το περιβάλλον του Eclipse (στις περιπτώσεις που είναι εφικτή η χρήση τους).

Το Visual Studio C++ 2008 Express βρίσκεται στην τοποθεσία:
http://msdn2.microsoft.com/en-us/express/future/bb421473.aspx
(Πρόσβαση: 10 Οκτωβρίου 2006)

Εικόνα 8 Visual Studio 2008 Express Editions Beta 2

Αφού ολοκληρωθεί η εγκατάσταση δημιουργούμε ένα νέο project: *File → New → Project. → Win32 → Win32 console application*, πληκτρολογούμε ένα όνομα για το project και στην συνεχεία πατάμε το *ok(Εικόνα 9)*.

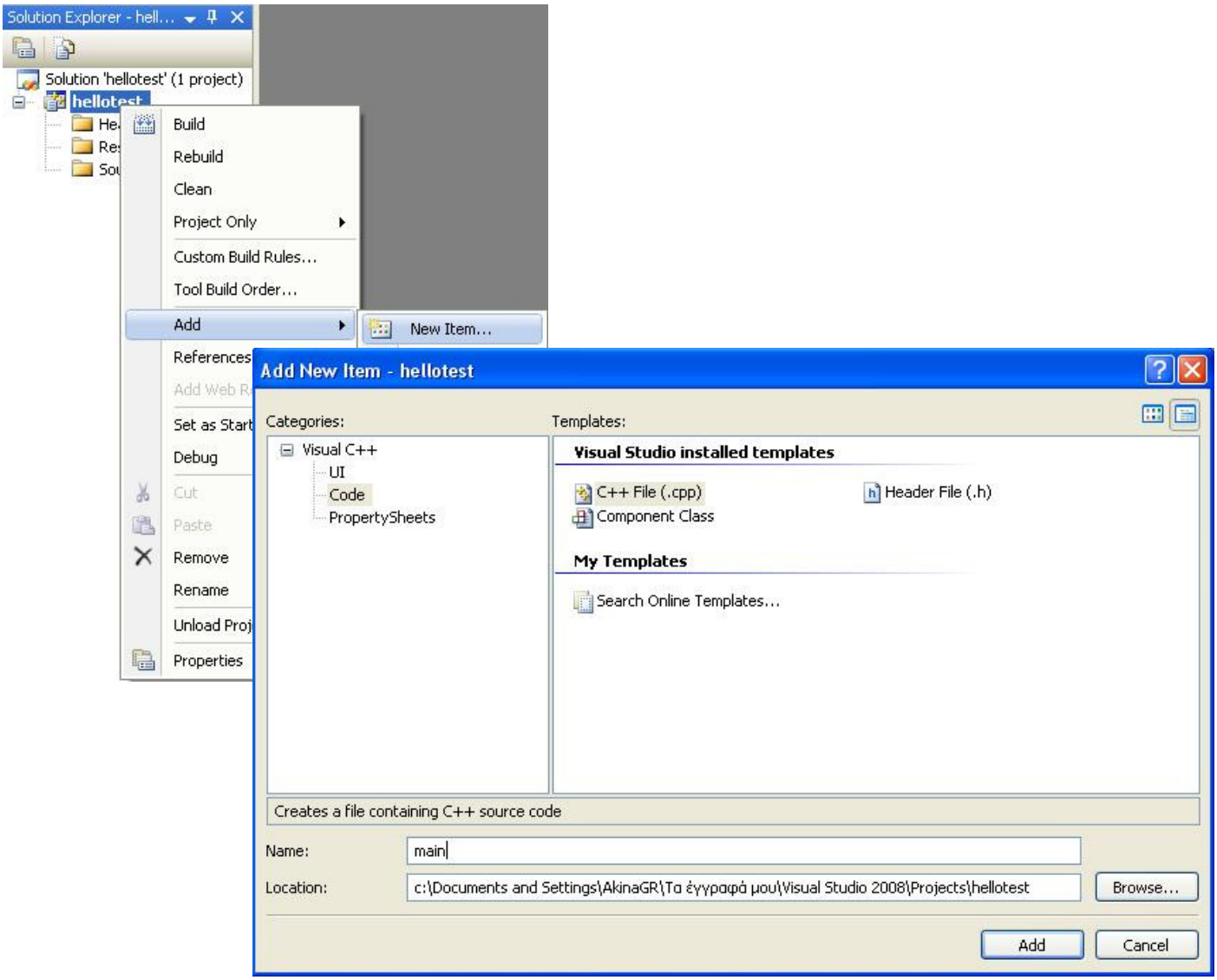


Εικόνα 9 Δημιουργία ενός νέου project.

Στην συνεχεία θα εμφανιστεί ένα *Win32 Application Wizard* παράθυρο, όπου επιλέγουμε *ok* και στην επόμενη καρτέλα: *'Console Application', 'Empty project'*, και τέλος, το *finish(Εικόνα 10).*

Εικόνα 10 Win32 Application Wizard.

Επιλέγοντας το όνομα του project που μόλις δημιουργήθηκε, δεξί *click->add->New items, visual C++ ->Code ->C++ file (.cpp)* πληκτρολογούμε ένα όνομα για το αρχείο κώδικα και στην συνεχεία πατάμε το *ok (Εικόνα 11)*.



Εικόνα 11 Δημιουργία ενός νέου *Source file*.

Στην συνέχεια εισάγουμε ένα απλό παράδειγμα κώδικα, σαν αυτό που φαίνεται παρακάτω:

```
#include <stdio.h>
int main(void) {
printf("Hello, world C!\n");
return 0;
}
```

Αποθηκεύουμε το αρχείο (ctrl+s) και στην συνεχεία πατάμε *build ->build solution(f7)*. Στο output του visual studio C++ πρέπει να εμφανιστεί το μήνυμα *«Build: 1 succeeded, 0 failed»*.

3. Εγκατάσταση του API OpenAL

Προκειμένου να λειτουργήσει οποιοδήποτε API, πρέπει να οριστούν στον μεταγλωττιστή που βρίσκονται τα αντίστοιχα Include, Library και Dynamic Link Library files.

Το API OpenAL παρέχεται μέσω ενός SDK, το οποίο βρίσκεται στη διεύθυνση:
http://www.openal.org/downloads.html (Πρόσβαση: 10 Οκτωβρίου 2006)

Εκτός από το SDK εκεί βρίσκεται και το ALUT, που είναι τμήμα της OpenAL και το οποίο στην τελευταία έκδοσή της, αποτελεί ξεχωριστό τμήμα από το API. Μέσα από τα Source αρχεία που διανέμονται, επίσης, στην ίδια σελίδα, μπορούμε να δούμε τον κώδικα που περιέχουν τα αρχεία llb και dll και ουσιαστικά να έχουμε άμεση πρόσβαση στα function της OpenAL για τυχόν τροποποιήσεις ή επεξηγήσεις στην λειτουργία της.

Κατά την εγκατάσταση του SDK θα χρησιμοποιήσουμε ως path: *C:\Program Files\ OpenAL 1.1 SDK*, όπου θα τοποθετήσουμε και την ALUT . Με την εγκατάσταση του SDK τα dll: *OpenAL32* και *wrap_oal.dll* τοποθετούνται στο default path *C:\WINDOWS\system32* και επομένως δεν χρειάζεται κάποια περαιτέρω διεργασία για τα Dynamic Link Library. Πρέπει να σημειωθεί εδώ, ότι για να τρέξει μια εφαρμογή OpenAL αυτά τα δύο αρχεία είναι απαραίτητα και επομένως πρέπει κάθε φορά να περιλαμβάνονται μαζί με το εκτελέσιμο αρχείο.

Στην συνεχεία κάνουμε Extract το αρχείο *Alut: freealut binary zip* και τοποθετούμε τα ALUT αρχεία ως εξής :

alut.h ⟶ *C:\Program Files\OpenAL1.1SDK\include*

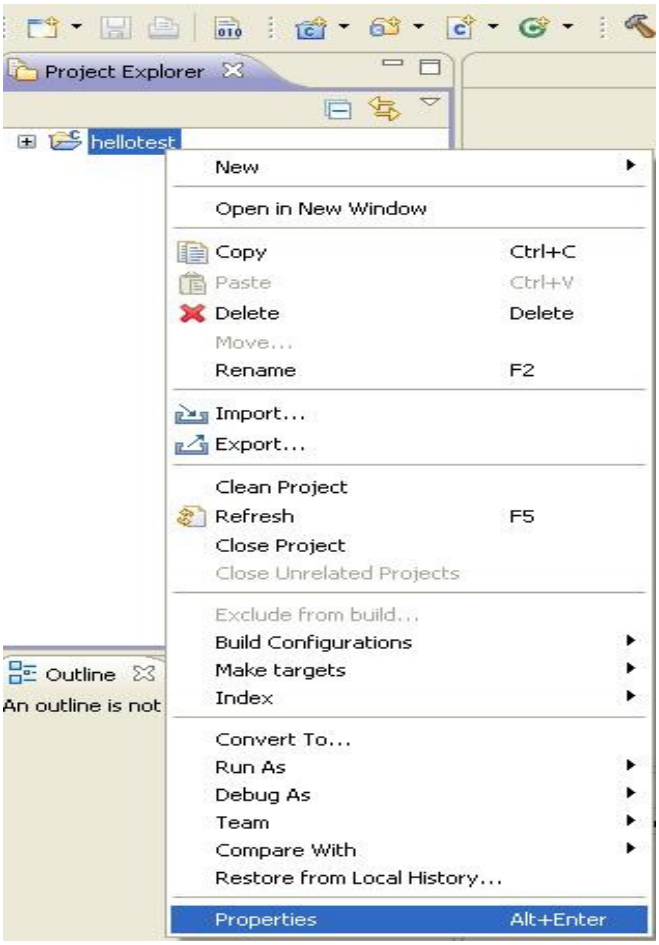
alut.lib ⟶ *C:\Program Files\OpenAL1.1SDK\libs\Win32*

alut.dll ⟶ *C:\WINDOWS\system32*

Σημειώνεται πως το αρχείο *alut.dll* πρέπει να περιλαμβάνεται μαζί με τα άλλα δύο dll αρχεία *(OpenAL32.dll, wrap_oal.dll).*

3.1 Eclipse & OpenAL

Επιλέγουμε δεξί click στο όνομα του Project, *properties(Alt +Enter).*



Εικόνα 12 Project properties.

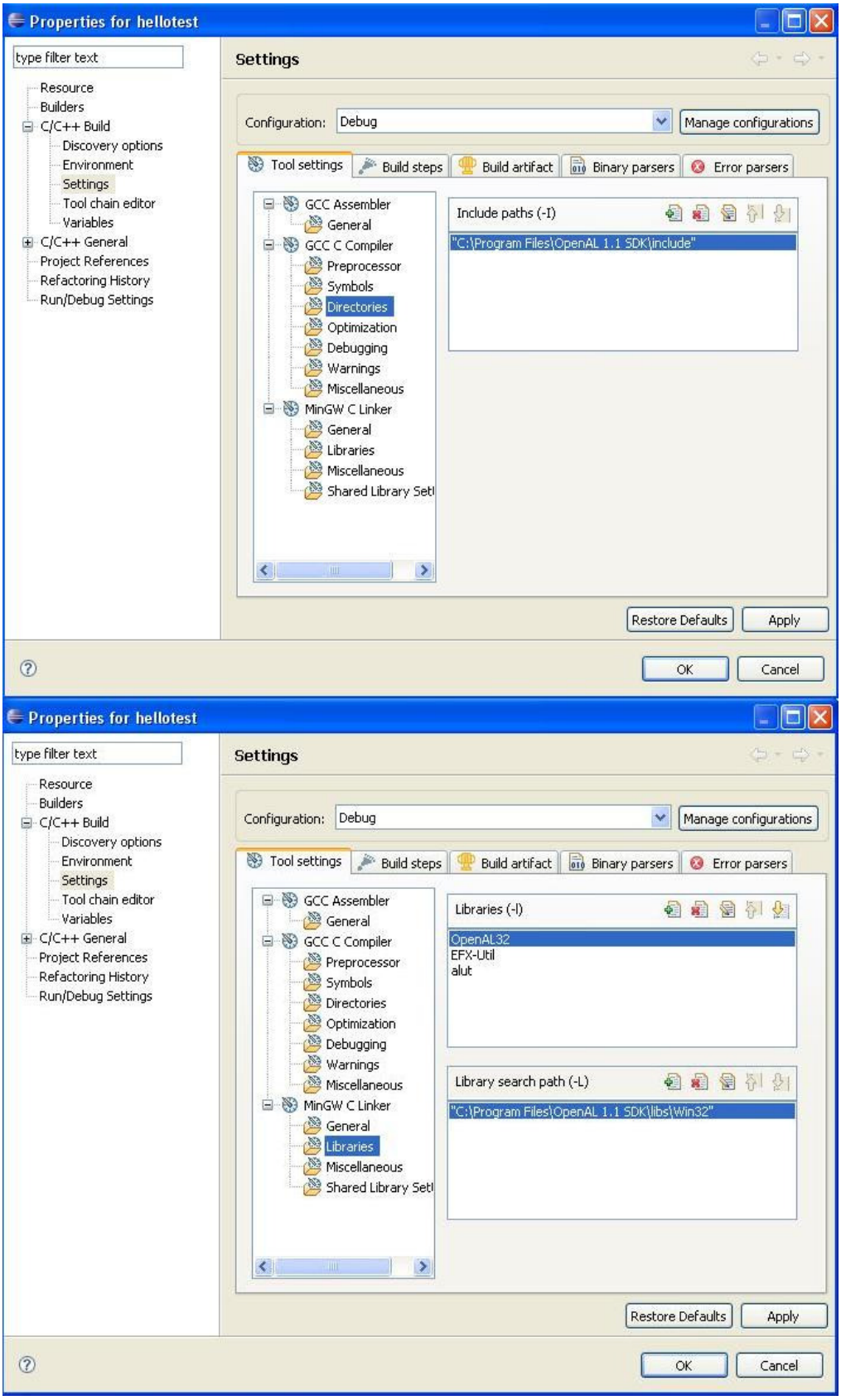
Στην συνεχεία C/C++ Build -> Settings και στις ενότητες που ακολουθούν εισάγουμε τα path για *Debug* και *Release(Εικόνα 13).*

- *GCC C Compiler –> Directories: (Προσθήκη στο Include Paths (-I))*
  C:\Program Files\OpenAL 1.1 SDK\include

- *GCC C Linker -> Libraries: (Προσθήκη στο libraries (-l))*

  OpenAL32, EFX-Util, alut

- *GCC C Linker -> Libraries: (Προσθήκη στο library search path (-L))*
  C:\Program Files\OpenAL 1.1 SDK\libs\Win32

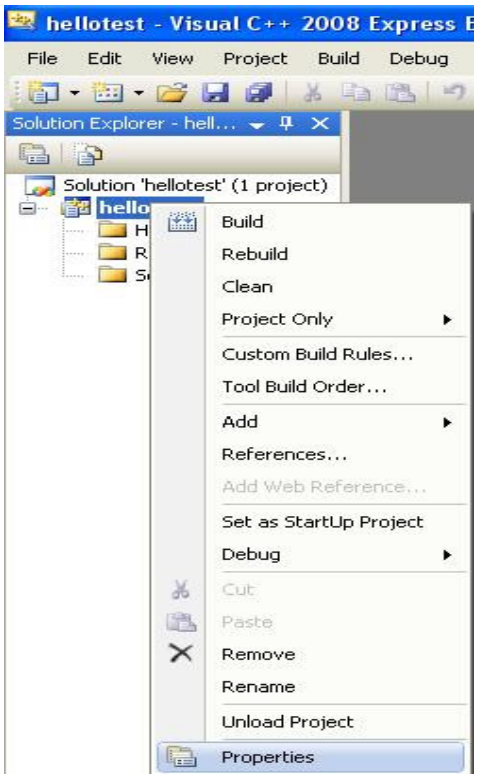


Εικόνα 13 *C/C++ Build*.

Πρέπει να σημειωθεί εδώ ότι μερικά από τα API που περιέχει το Microsoft Platform SDK μπορούν να λειτουργήσουν και στο Eclipse με τον ίδιο ακριβώς τρόπο. Ένα παράδειγμα είναι το lib winmm.lib που χρησιμοποιείται από κάποια παραδείγματα του SDK της OpenAL και περιέχει τις εντολές waveInGetNumDevs και waveOutGetNumDevs. Το path του συγκεκριμένου lib είναι : C:\Program Files\ Microsoft Platform SDK\Lib.

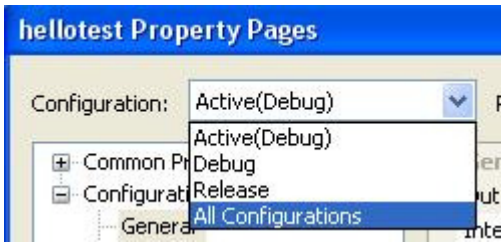
3.2 Visual Express Edition & OpenAL

Αντίστοιχα και στο Visual Express Edition πρέπει να δηλωθούν τα αρχεία που δηλώθηκαν στο Eclipse. Επιλέγουμε το όνομα του project, *δεξί click-> Properties* (Εικόνα 14).



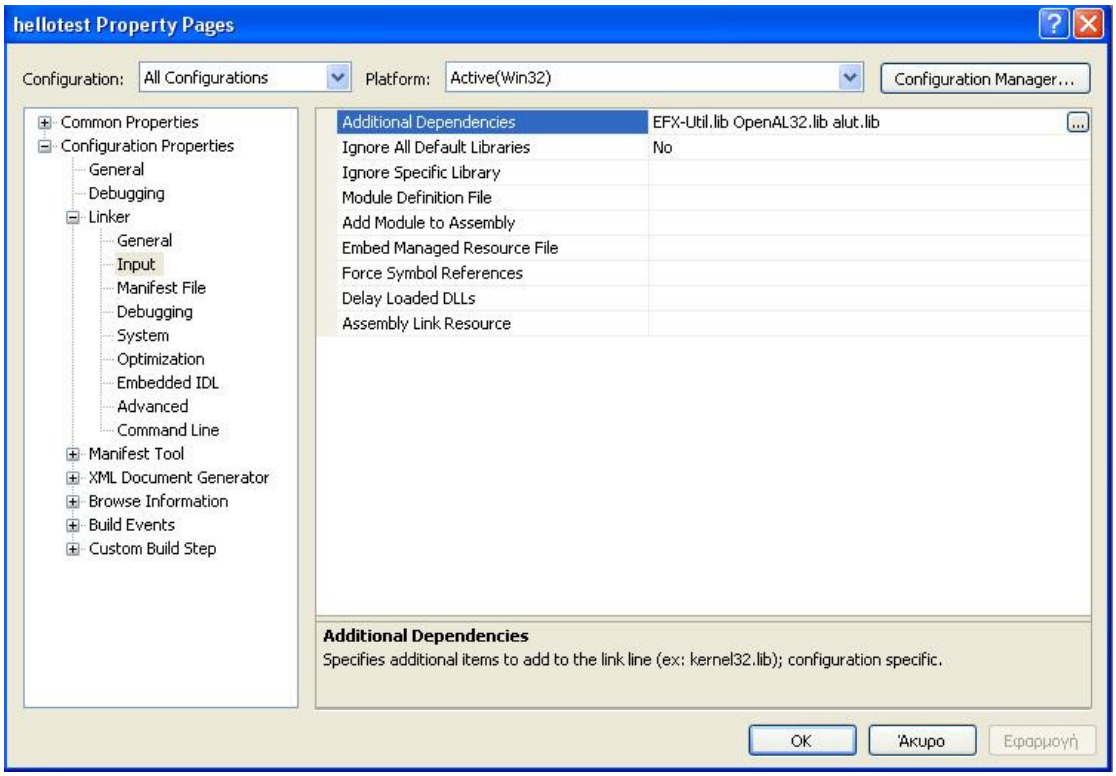
Εικόνα 14 Visual Express Edition property.

Στην καρτέλα property page που θα εμφανισθεί, πηγαίνουμε στο *Configuration box* και επιλέγουμε *All Configurations,* ώστε οι ρυθμίσεις που θα κάνουμε να ισχύουν και για *Debug* και για *Release*.



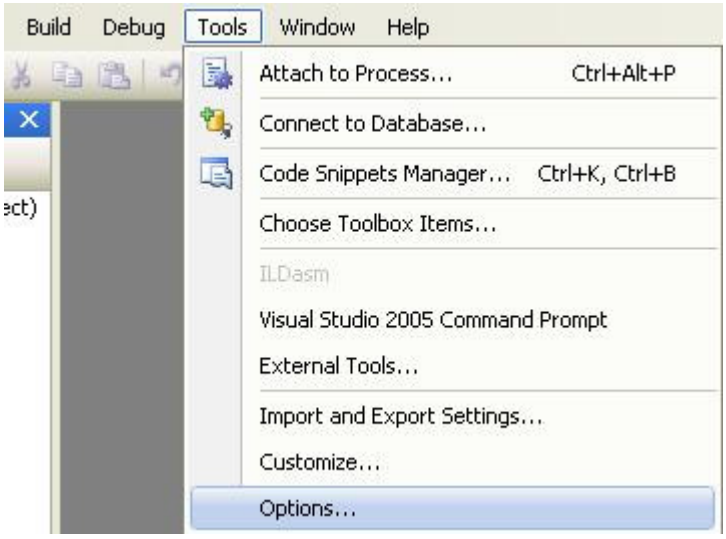
Εικόνα 15 *All Configurations.*

Στην συνέχεια επιλέγουμε *Linker-> Input* και εισάγουμε τα ακόλουθα lib στο πεδίο *Additional Dependencies : EFX-Util.lib OpenAL32.lib alut.lib* (Εικόνα 16)



Εικόνα 16 *Additional Dependencies*

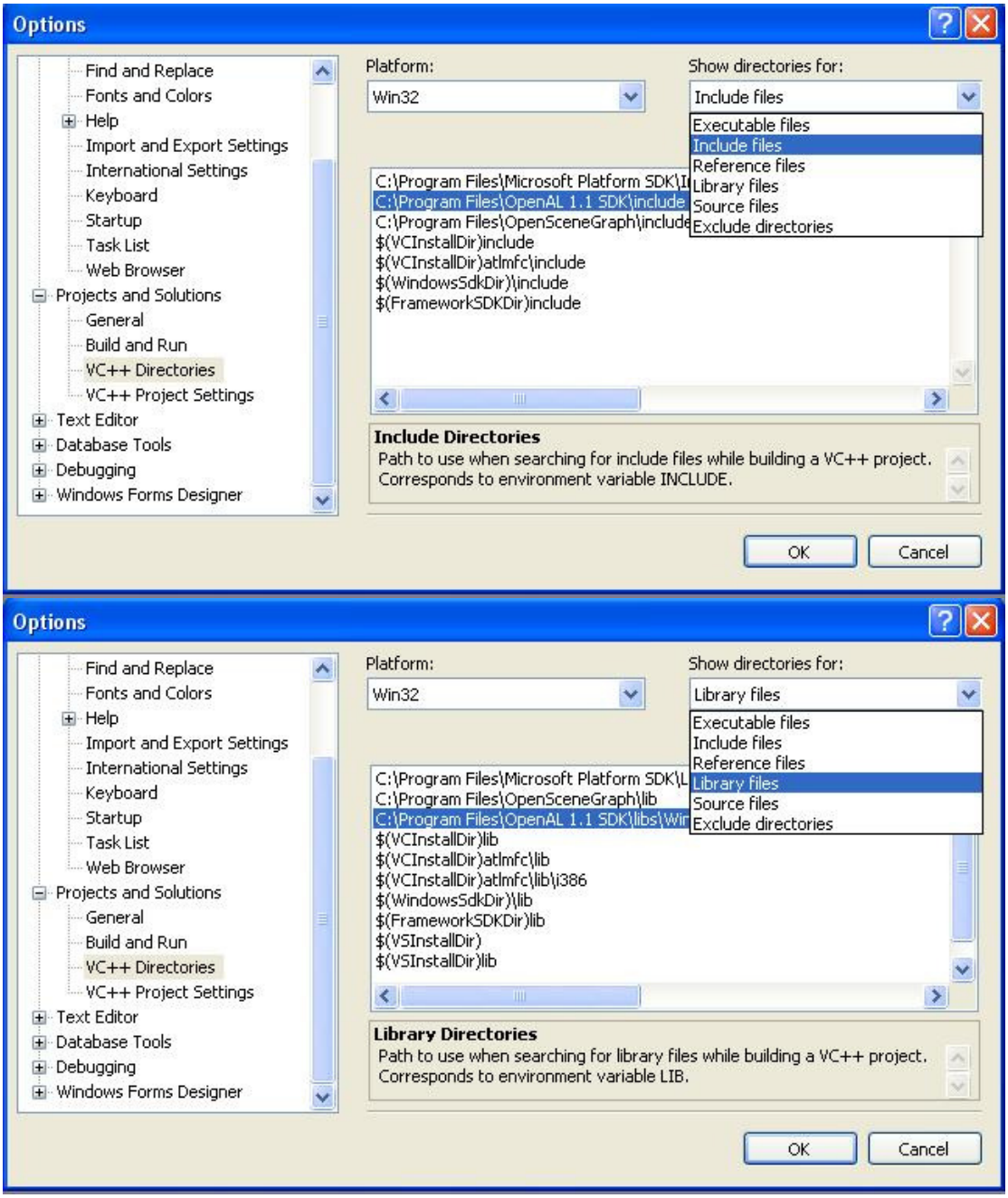
Επίσης πηγαίνουμε στα *tools->Options*



Εικόνα 17 Options..

Στην συνεχεία, στην καρτέλα Options που θα εμφανισθεί, πηγαίνουμε στο *Projects and Solutions->VC++ Directories* και εισάγουμε (Εικόνα 18) :

- Include files: C:\Program Files\OpenAL 1.1 SDK\include
- Library files: C:\Program Files\OpenAL 1.1 SDK\libs\Win32



Εικόνα 18 *VC++ Directories.*

Πρέπει να σημειωθεί ότι οι εφαρμογές που δημιουργούνται μέσω του Visual Express Edition C++, εκτός από τα dll της OpenAL (*OpenAL32.dll, wrap_oal.dll alut.dll*), που πρέπει να συμπεριληφθούν, ώστε να λειτουργήσει σωστά η εφαρμογή, χρειάζονται και τα dll του ίδιου του Visual Express Edition C++, τα οποία παρέχονται με σχετική άδεια χρήσης από το ίδιο το Visual Express Edition C++. Στην περίπτωση που η εφαρμογή είναι CRT(default), αυτά είναι τα εξής: *Microsoft.VC80.CRT.manifest, msvcm80.dll, msvcp80.dll* και *msvcr80.dll*.

Η πλήρης λίστα των αρχείων, που έχουν σχετική άδεια, βρίσκεται στον δικτυακό τόπο:
http://download.microsoft.com/download/b/d/6/bd6d539a-e2c3-49d0-97ad-443492c79122/VS2008Beta2Redist.htm (Πρόσβαση: 10 Οκτωβρίου 2006)

ΑΝΑΦΟΡΕΣ

1. Audio Software Interfaces :
http://users.teicrete.gr/taxd/02/notes/driverapis/31.APIs%20and%20drivers.htm
(Πρόσβαση : 20 Σεπτεμβρίου 2006)

2. Device Driver :
http://en.wikipedia.org/wiki/Device_driver (Πρόσβαση : 20 Σεπτεμβρίου 2006)

3. API:
http://www.webopedia.com/TERM/A/API.html (Πρόσβαση : 20 Σεπτεμβρίου 2006)

4. Application Programming Interface :
http://en.wikipedia.org/wiki/API (Πρόσβαση : 20 Σεπτεμβρίου 2006)

5. Application Programming Interface API :
http://www.freesoft.org/CIE/Topics/3.htm (Πρόσβαση : 20 Σεπτεμβρίου 2006)

6. Application Programming Interface :
http://www.answers.com/topic/application-programming-interface (Πρόσβαση : 4 Οκτωβρίου 2006)

7. Free Software :
http://en.wikipedia.org/wiki/Free_software (Πρόσβαση : 4 Οκτωβρίου 2006)

8. Software Development Kit – SDK :
http://en.wikipedia.org/wiki/Software_development_kit (Πρόσβαση : 4 Οκτωβρίου 2006)

9. SDK :
http://www.webopedia.com/TERM/S/SDK.html (Πρόσβαση : 4 Οκτωβρίου 2006)

10. Software Framework :
http://en.wikipedia.org/wiki/Software_framework (Πρόσβαση : 4 Οκτωβρίου 2006)

11. OpenGL Programming In Eclipse :
http://www.ferdychristant.com/blog/articles/DOMM-72MPPE (Πρόσβαση : 23 Οκτωβρίου 2007)

12. Embedded System :
http://en.wikipedia.org/wiki/Embedded_system (Πρόσβαση : 4 Οκτωβρίου 2006)

13. IDE:
http://en.wikipedia.org/wiki/Integrated_development_environment (Πρόσβαση : 4 Οκτωβρίου 2006)

14. Header File :
http://en.wikipedia.org/wiki/Header_file (Πρόσβαση : 23 Οκτωβρίου 2007)

15. Library(Computering) :
http://en.wikipedia.org/wiki/Library_%28computer_science%29 (Πρόσβαση : 23 Οκτωβρίου 2007)

16. Dynamic Ling Library :
http://en.wikipedia.org/wiki/Dynamic_Link_Library (Πρόσβαση : 23 Οκτωβρίου 2007)

17. Programming 3d Sound With Openal In Windows:
http://www.gamedev.net/reference/articles/article1958.asp (Πρόσβαση : 4 Οκτωβρίου 2006)

18. DirectSound :
http://en.wikipedia.org/wiki/DirectSound#DirectSound3D (Πρόσβαση : 15 Οκτωβρίου 2006)

19. DirectX SDK : http://msdn2.microsoft.com/en-us/xna/aa937781.aspx (Πρόσβαση : 15 Οκτωβρίου 2006)

20. PortAudio :
http://www.portaudio.com/ (Πρόσβαση : 20 Οκτωβρίου 2006)

21. PortAudio Tutorial Code :
http://www.portaudio.com/docs/portaudio_h.txt (Πρόσβαση : 20 Οκτωβρίου 2006)

22. AudioMulch :
http://www.audiomulch.com/ (Πρόσβαση : 27 Φεβρουαρίου 2008)

23. JSyn :
http://www.softsynth.com/jsyn/ (Πρόσβαση : 27 Φεβρουαρίου 2008)

24. PortAudio Tutorial Code :
http://www.portaudio.com/docs/portaudio_h.txt (Πρόσβαση : 20 Οκτωβρίου 2006)

25. Portaudio –An Open Source Crossplatrfom Audio Api :
http://www.audiomulch.com/~rossb/writings/portaudio_icmc2001.pdf (Πρόσβαση : 20 Οκτωβρίου 2006)

26. Libsndfile :
http://www.mega-nerd.com/libsndfile/ (Πρόσβαση : 20 Οκτωβρίου 2006)

27. Bass :
http://www.un4seen.com/ (Πρόσβαση : 25 Οκτωβρίου 2006)

28. Free Audio / Sound Libraries And Source Code :
http://www.thefreecountry.com/sourcecode/audio.shtml (Πρόσβαση : 25 Οκτωβρίου 2006)

29. A3D :
http://en.wikipedia.org/wiki/A3D (Πρόσβαση : 25 Οκτωβρίου 2006)

30. Mod(File Format) :
http://en.wikipedia.org/wiki/MOD_(file_format) (Πρόσβαση : 10 Νοεμβρίου 2007)

31. MOD-FAQ part 1 of 2:
http://www.koeln.netsurf.de/~michael.mey/faq1.txt (Πρόσβαση : 10 Νοεμβρίου 2007)

32. Programming 3d Sound With Openal In Windows :
http://www.gamedev.net/reference/articles/article1958.asp (Πρόσβαση : 31 Οκτωβρίου 2006)

33. OpenAL Lesson 1 - Simple Static Sound :
http://www.devmaster.net/articles/openal-tutorials/lesson1.php (Πρόσβαση : 31 Οκτωβρίου 2006)

34. OpenAL:
http://www.openal.org/ (Πρόσβαση : 31 Οκτωβρίου 2006)

35. OpenAL 1.1 SDK – OpenAL 1.1 Specifications. PDF

36. OpenAL 1.1 SDK - OpenAL Programmer's Guide. PDF

37. Interactive Audio Special Interest Group :
http://www.iasig.org/ (Πρόσβαση : 20 Νοεμβρίου 2007)

38. Ήχος Στα Vista – Αποχαιρετίστε Το Directsound3d Και Υποδεχτείτε Το Openal : Περιοδικό RAM, Τεύχος 213 - Μάιος 2007, σελ.156

39. Χωρίς DirectSound 3d Hardware Acceleration Τα Vista :
http://wiggler.gr/2007/01/15/vista-not-support-directsound-3d-eax/ (Πρόσβαση : 3 Απριλίου 2007)

40. ASIO – Audio Stream Input/Output :
http://en.wikipedia.org/wiki/Audio_stream_input_output (Πρόσβαση : 5 Σεπτεμβρίου 2007)

41. Vista Tips Και Κόλπα Για Όλους :
http://www.freegr.gr/freenuke/modules.php?name=News&file=article&sid=200
(Πρόσβαση : 3 Απριλίου 2007)

42. Loki Entertainment Software :
http://www.lokigames.com/ (Πρόσβαση : 25 Σεπτεμβρίου 2007)

43. OpenAL Title :
http://www.openal.org/titles.html (Πρόσβαση : 25 Σεπτεμβρίου 2007)

44. OpenAL Link :
http://www.openal.org/links.html (Πρόσβαση : 25 Σεπτεμβρίου 2007)

45. Alut Specification :
http://www.openal.org/openal_webstf/specs/alut.html (Πρόσβαση : 28 Σεπτεμβρίου 2007)

46. Context(Computing) :
http://en.wikipedia.org/wiki/Context_%28computing%29(Πρόσβαση : 26 Νοεμβρίου 2007)

47. BOOK 3D Game Programming - 3D Game Programming Al in One (Finney 2004).pdf

48. Velocity :
http://en.wikipedia.org/wiki/Velocity (Πρόσβαση : 29 Νοεμβρίου 2007)

49. Chapter N13. 3D Sound with JOAL :
http://fivedots.coe.psu.ac.th/~ad/jg2/ch13/joal.pdf (Πρόσβαση : 10 Δεκεμβρίου 2007)

50. Listener Position And Orientation :
http://opensource.creative.com/pipermail/openal/2004-November/007887.html
(Πρόσβαση : 13 Δεκεμβρίου 2007)

51. GameDev.net - Programming 3D Sound With OpenAL in Windows :
http://www.gamedev.net/reference/articles/article1958.asp (Πρόσβαση : 13 Δεκεμβρίου 2007)

52. NeHe Productions: OpenGL Article #19 :
http://nehe.gamedev.net/data/articles/article.asp?article=19 (Πρόσβαση : 10 Δεκεμβρίου 2007)

53. VirtualCamera :
http://nccastaff.bournemouth.ac.uk/jmacey/ProgGraph/slides/VirtualCamera.pdf
(Πρόσβαση : 15 Δεκεμβρίου 2007)

54. OpenGL– OpenGL 2.1 Specifications.pdf :
http://www.opengl.org/registry/doc/glspec21.20061201.pdf (Πρόσβαση : 15 Δεκεμβρίου 2007)

55. OpenGL Camera :
http://www.morrowland.com/apron/article/gl/camera/index.php (Πρόσβαση : 15 Δεκεμβρίου 2007)

56. OpenGL Matrix :
http://www.morrowland.com/apron/tutorials/gl/gl_matrix.php (Πρόσβαση : 15 Δεκεμβρίου 2007)

57. Virtual reality :
http://en.wikipedia.org/wiki/Virtual_reality (Πρόσβαση : 25 Δεκεμβρίου 2007)

58. OpenAL 1.1 SDK - Effects Extension Guide. PDF

59. Εγχειρίδιο ΑΚΟΥΣΤΙΚΗΣ –F. Alton Everest 3η Έκδοση

60. EAX :
http://en.wikipedia.org/wiki/Environmental_audio_extensions (Πρόσβαση : 6 Ιανουαρίου 2008)

61. SoundBlaster EAX :
http://www.soundblaster.com/eax/abouteax/ (Πρόσβαση : 6 Ιανουαρίου 2008)

62. Integrated Development Environment :
http://en.wikipedia.org/wiki/integrated_development_environment (Πρόσβαση: 10 Οκτωβρίου 2006)

63. Eclipse :
http://www.eclipse.org/ (Πρόσβαση: 10 Οκτωβρίου 2006)

64. Eclipse For The Versatile Developer :
http://www.ferdychristant.com/blog/articles/domm-6rge3d (Πρόσβαση: 10 Οκτωβρίου 2006)

65. Gcc, The Gnu Compiler Collection :
http://gcc.gnu.org/ (Πρόσβαση: 10 Οκτωβρίου 2006)

66. MinGW :
http://www.mingw.org/ (Πρόσβαση: 10 Οκτωβρίου 2006)

67. Glut Setup Tutorial With Eclipse Cdt On Windows :
http://www.ritgamedev.com/tutorials/gluteclipse/(Πρόσβαση: 10 Οκτωβρίου 2006)

68. Ogre3d SDK Support In Eclipse :
http://www.ogre3d.org/wiki/index.php/eclipse_mingw (Πρόσβαση: 10 Οκτωβρίου 2006)

69. GccWinBinaries :
http://www.develer.com/oss/gccwinbinaries (Πρόσβαση: 10 Οκτωβρίου 2006)

70. Installing An SDK :
http://www.ogre3d.org/wiki/index.php/installing_an_sdk (Πρόσβαση: 16 Οκτωβρίου 2006)

71. Installing An SDK :
http://www.ogre3d.org/wiki/index.php/installing_an_sdk (Πρόσβαση: 16 Οκτωβρίου 2006)