ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ
ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

«ΔΙΑΔΙΚΤΥΑΚΗ ΕΦΑΡΜΟΓΗ 2D ΚΑΙ 3D ΑΡΧΙΤΕΚΤΟΝΙΚΟΥ ΣΧΕΔΙΑΣΜΟΥ ΒΑΣΙΣΜΕΝΗ ΣΕ ΤΕΧΝΟΛΟΓΙΕΣ SVG, X3D ΚΑΙ AJAX»

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΚΑΓΚΕΛΙΔΗΣ ΚΩΝΣΤΑΝΤΙΝΟΣ

Εξεταστική Επιτροπή

Επ.Καθ. Αικατερίνη Μανιά
Επ.Καθ. Αντώνιος Δεληγιαννάκης
Αν.Καθ. Αλέξανδρος Ποταμιάνος

ΧΑΝΙΑ 2010

στην οικογένειά μου

Περίληψη

Ως αντικείμενο της παρούσας εργασίας, δημιουργήθηκε μια εφαρμογή αρχιτεκτονικής σχεδίασης και διακόσμησης χώρων. Η εφαρμογή αυτή έχει διαδικτυακή μορφή, δηλαδή εκτελείται μέσο περιβάλλον φυλλομετρητή, με την μορφή ιστοσελίδων με υψηλή διαδραστικότητα. Η εφαρμογή επιτρέπει στον χρήστη να δημιουργήσει δισδιάστατα σχέδια κατόψεων (floor plans) τα οποία μπορεί να διακοσμήσει με αντικείμενα (έπιπλα, ηλ. συσκευές κτλ). Παρέχεται μια μεγάλη γκάμα εργαλείων σχεδίασης. Δίνεται η δυνατότητα στον χρήστη να οριοθετήσει δομικά στοιχεία του σχεδίου, όπως τοίχοι, δωμάτια, οροφές κτλ. Από ειδικές παλέτες μπορεί να επιλέξει τύπους παραθύρων και πορτών για να τα τοποθετήσει στους τοίχους. Παρέχονται εργαλεία για την απόδοση υφών στα επιφάνειες, όπως επιλογή πλακιδίων για τα δάπεδα, επιλογή ταπετσαρίας για τους τοίχους. Αφού ολοκληρωθεί η σχεδίαση των χώρων, ο χρήστης μπορεί να χρησιμοποιήσει αντικείμενα από μια βάση δεδομένων για να τους διακοσμήσει. Η εφαρμογή παρέχει εργαλεία τοποθέτησης και περιστροφής τον αντικειμένων. Επίσης υπάρχει επιλογή για την εμφάνιση οδηγών όπως μορφή πλέγματος στην επιφάνεια σχεδίασης, καθώς και πληροφόρηση για τις διαστάσεις των τοίχων. Κατόπιν η εφαρμογή δημιουργεί μια τρισδιάστατη αναπαράσταση των χώρων που σχεδίασε ο χρήστης και επιτρέπει την περιήγησή του σε αυτούς. Κατά την περιήγηση δίνεται δυνατότητα αλληλεπίδρασης με διάφορα αντικείμενα. Επίσης περιλαμβάνει την υλοποίηση ενός ισχυρού επιλογέα χρωμάτων. Πιο συγκεκριμένα, δίνεται η δυνατότητα στον χρήστη να επιλέγει ταυτόχρονα περισσότερα του ενός χρώματα τα οποία διαθέτουν χρωματική αρμονία. Αυτό επιτυγχάνεται εφαρμόζοντας κανόνες επιλογής χρωμάτων όπως ο αναλογικός, ο τριαδικός και ο συμπληρωματικός. Τέλος δίνεται η επιλογή για αποθήκευση και ανάκτηση των σχεδίων κατόψεων που δημιουργεί ο χρήστης. Επίσης υπάρχει δυνατότητα επισκόπησης σχεδίων άλλων χρηστών. Κατά τον σχεδιασμό της εφαρμογής μελετήθηκε ένας μεγάλος αριθμός τεχνολογιών για την ανάπτυξη των διάφορων τμημάτων. Όσον αφορά την υλοποίηση, η εφαρμογή, αποτελείται από ένα τμήμα το οποίο εκτελείται στο περιβάλλον του φυλλομετρητή του χρήστη (client) και ένα τμήμα το οποίο εκτελείται σε περιβάλλον διακομιστή ιστού (web server). Επίσης χρησιμοποιείται βάση δεδομένων για διάφορα δεδομένα τα οποία πρέπει να αποθηκευτούν και να ανακτηθούν. Για την υλοποίηση έγινε χρήση σύγχρονων τεχνολογιών ανάπτυξης διαδικτυακών εφαρμογών, στις οποίες συγκαταλέγονται οι εξής: HTML για την δόμηση των ιστοσελίδων της εφαρμογής, Javascript για την συγγραφή κώδικα που εκτελείται στον φυλλομετρητή του χρήστη, PHP για την συγγραφή κώδικα που εκτελείται στον διακομιστή ιστού, MySql για την υποστήριξη της βάσης δεδομένων, SVG ως γλώσσα αναπαράστασης δισδιάστατων διανυσματικών γραφικών και X3D ως γλώσσα αναπαράστασης τρισδιάστατων

γραφικών. Σημειώνεται ότι η Javascript χρησιμοποιείται για να προσδώσει διαδραστικότητα, τόσο στα έγγραφα HTML τα οποία αποτελούν την γραφική διεπαφή του χρήστη, όσο και στα γραφικά στοιχεία της SVG κατά την σχεδίαση των κατόψεων. Τα σχέδια των χρηστών αποθηκεύονται στον διακομιστή με μορφή κειμένου σε γλώσσα XML. Η ανάκτηση/αποθήκευση δεδομένων απαιτούν επικοινωνία με τον διακομιστή. Η επικοινωνία αυτή γίνεται με τρόπο ασύγχρονο, μέσω Javascript. Χρησιμοποιείται το αντικείμενο XMLHttpRequest το οποίο αναλαμβάνει την αποστολή αιτήσεων και παραλαβή απαντήσεων από τον διακομιστή ιστού. Η αιτήσεις αυτές διεκπεραιώνονται από λειτουργία κατάλληλων διαδικασιών σε php. Η php επικοινωνεί με την βάση δεδομένων του διακομιστή και δημιουργεί απαντήσεις σε μορφή XML της οποίες στέλνει πίσω στην javascript. Αυτή η ασύγχρονη επικοινωνία javascript και διακομιστή XML ονομάζεται AJAX (Asynchronous Javascript and XML). Η εφαρμογή επίσης υποστηρίζει την εγγραφή χρηστών και δημιουργία προφίλ. Επίσης δίνει την δυνατότητα ανταλλαγής σχολίων μεταξύ των χρηστών. Η εφαρμογή υλοποιήθηκε σε περιβάλλον Windows 7 χρησιμοποιώντας τοπικά τον διακομιστή ιστού Apache Http Server.

Ευχαριστίες

Θα ήθελα να εκφράσω τις ιδιαίτερες ευχαριστίες μου στην επιβλέπουσα καθηγήτρια μου κ. Κατερίνα Μανιά, τόσο στα πλαίσια αυτής της διπλωματικής εργασίας, για την επίβλεψη, την καθοδήγηση και την ανεκτίμητη βοήθεια που παρείχε, όσο και στα πλαίσια του μαθήματός της «Γραφική», το οποίο αποτέλεσε κομβικό σημείο για την γνωριμία μου με το συναρπαστικό αντικείμενο της αναπαράστασης γραφικών με την βοήθεια υπολογιστή. Επίσης θα ήθελα να ευχαριστήσω τους καθηγητές κ. Ποταμιάνο Αλέξανδρο και κ. Δεληγιαννάκη Αντώνιο, για το χρόνο που θα αφιερώσουν στην ανάγνωση του κειμένου και για τις εποικοδομητικές παρατηρήσεις τους. Θα ήθελα να ευχαριστήσω επίσης όλους όσους συμμετείχαν στην επαναληπτική διαδικασία χρήσης της εφαρμογής, οι οποίοι, από τα πρώτα κιόλας στάδια, βοήθησαν με τις παρατηρήσεις και αξιολογήσεις στην βελτίωση της.

Ένα τεράστιο ευχαριστώ, χρωστώ στην οικογένεια μου για την αμέριστη και απόλυτη υποστήριξη που μου παρείχε και συνεχίζει να μου παρέχει.

Τέλος, ένα μεγάλο ευχαριστώ στους ανθρώπους εκείνους που νιώθω κοντά μου, για την κατανόηση, την αγάπη, το κουράγιο και την έμπνευση που μου δίνουν τόσο στις καλές όσο και στις δύσκολες στιγμές.

Σεπτέμβριος 2009

Κωνσταντίνος Καγκελίδης

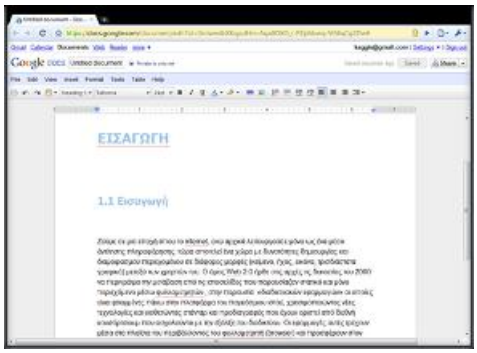
Περιεχόμενα

Ευρετήριο Εικόνων

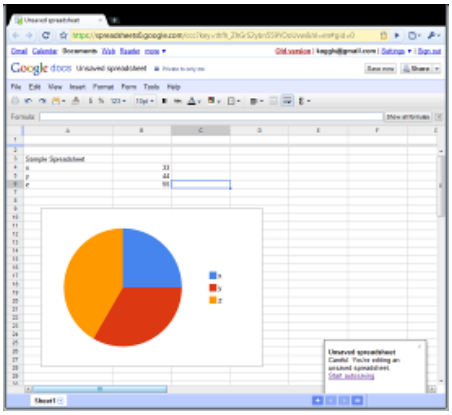
ΚΕΦΑΛΑΙΟ 1
ΕΙΣΑΓΩΓΗ

1.1 Εισαγωγή

Ζούμε σε μια εποχή όπου το internet, ενώ αρχικά λειτουργούσε μόνο ως ένα μέσο άντλησης πληροφόρησης, τώρα αποτελεί ένα χώρο με δυνατότητες δημιουργίας και διαμοιρασμού περιεχομένου σε διάφορες μορφές (κείμενο, ήχος, εικόνα, τρισδιάστατα γραφικά). Ο όρος Web 2.0 ήρθε, στις αρχές τις δεκαετίας του 2000, για να περιγράψει την μετάβαση από τις ιστοσελίδες που παρουσίαζαν στατικό και μόνο περιεχόμενο, μέσω φυλλομετρητών, στην παρουσία «διαδικτυακών εφαρμογών», οι οποίες είναι φτιαγμένες πάνω στην πλατφόρμα του παγκόσμιου ιστού, χρησιμοποιώντας νέες τεχνολογίες και υιοθετώντας στάνταρ και προδιαγραφές, που έχουν οριστεί από διεθνή κονσόρτσιουμ που ασχολούνται με την εξέλιξη του διαδικτύου. Οι εφαρμογές αυτές, τρέχουν μέσα στα πλαίσια του περιβάλλοντος του φυλλομετρητή (browser) και προσφέρουν στον χρήστη υψηλή διαδραστικότητα με τα αντικείμενα της ιστοσελίδας, συνήθως με σκοπό την δημιουργία περιεχομένου. Οι δυνατότητες των εφαρμογών αυτών, αν και ήταν περιορισμένες αρχικά, με την πάροδο των ετών έχουν πλησιάσει κατά πολύ τις δυνατότητες τον desktop εφαρμογών. Παραδείγματα τέτοιον πλούσιων εφαρμογών, αποτελούν ιστοσελίδες με μεγάλη διαδραστικότητα όπως τα Google Docs (docs.google.com), τα οποία μέσω του ιστοτόπου τους υλοποιούν μια σουίτα εφαρμογών γραφείου που εκτελούνται αποκλειστικά μέσα από το περιβάλλον του φυλλομετρητή, χωρίς την ύπαρξη plugin (εικόνες 1 και 2).

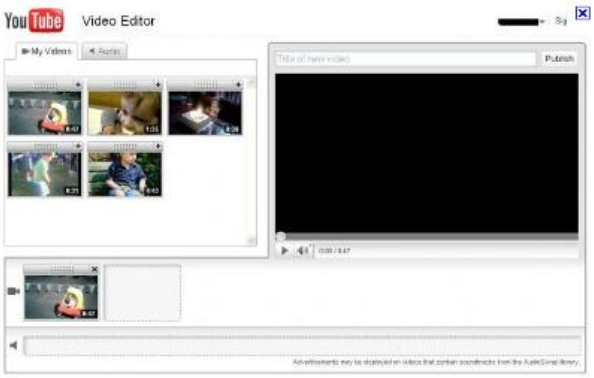


Εικόνα 1 – Το έγγραφο κειμένου της παρούσας διπλωματικής εργασίας ανοιγμένο μέσα στον διαδικτυακό επεξεργαστή κειμένου της goggle.

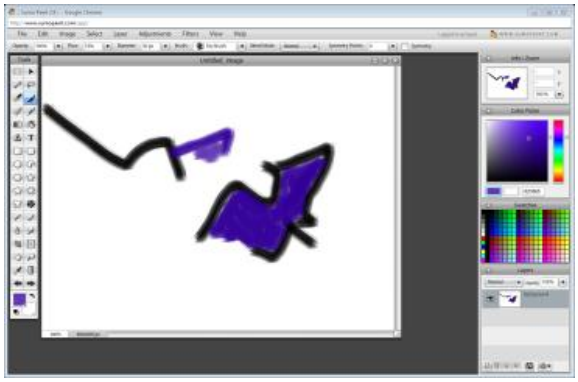


Εικόνα 2 - Επεξεργασία λογιστικού φύλλου μέσα σε περιβάλλον φυλλομετρητή χρησιμοποιώντας την ιστοσελίδα των goggle docs.

Οι εφαρμογές οι οποίες εντοπίζονται σήμερα σε διάφορες ιστοσελίδες, δεν περιορίζονται μόνο σε επεξεργαστές κειμένου ή λογιστικά φύλλα. Με την εξέλιξη των τεχνολογιών και την όλο και αυστηρότερη οριοθέτηση και τήρηση στάνταρτ στην κατασκευή και λειτουργία ιστοσελίδων, προέκυψε η δυνατότητα για την δημιουργία πολυμεσικών εφαρμογών στο διαδίκτυο, όπως online επεξεργαστές εικόνας, ήχου και βίντεο (εικόνες 3, 4 και 5).



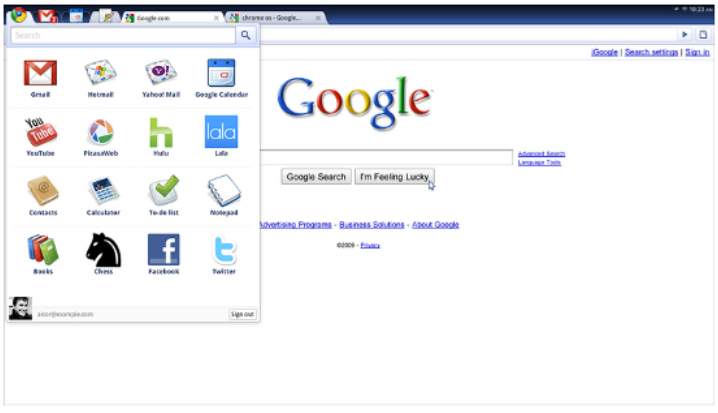
Εικόνα 3 - Ο online video editor της πασίγνωστης διαδικτυακής υπηρεσίας youtube.



Εικόνα 4 - Online πρόγραμμα επεξεργασίας εικόνας sumopaint.

Εικόνα 5 - Online πρόγραμμα επεξεργασίας ήχου aviary.

Έτσι, ειδικά προς τα τέλη της δεκαετίας του ’00, παρατηρούμε μια μεγάλη τάση για μεταφορά των εφαρμογών από την επιφάνεια εργασίας του υπολογιστή μας, στον ιστό χρησιμοποιώντας, σαν παράθυρο πρόσβασης σε αυτές, τον εκάστοτε φυλλομετρητή της επιλογής μας. Θα μπορούσαμε αναλογικά να πούμε ότι, ότι αποτελούσε η επιφάνεια εργασιών (desktop) των λειτουργικών συστημάτων για τις κλασσικές desktop εφαρμογές, αποτελεί τώρα ο φυλλομετρητής (browser) για τις web applications. Φυσικά η δημιουργία περιεχομένου αποθηκεύεται στο «σύννεφο» των εξυπηρετητών του ιστού καθιστώντας την προσβάσιμη από οποιοδήποτε σημείο και ιντερνετική συσκευή. Αυτό συνεισφέρει και στον εύκολο διαμοιρασμό περιεχομένου. Ένα σημάδι αυτής της μετάβασης, αποτελεί η ανακοίνωση του Chrome OS της Google, το οποίο έχει σχεδιαστεί ώστε ολόκληρο το λειτουργικό σύστημα να είναι ένας φυλλομετρητής και οι εφαρμογές του να είναι αποκλειστικά διαδικτυακές (εικόνες 6 και 7).



Εικόνα 6 - Στιγμιότυπο από το λειτουργικό σύστημα της Google, Chrome Os του οποίου ολόκληρη η διεπαφή είναι ένας φυλλομετρητής και οι εφαρμογές του είναι web applications που παρουσιάζονται φορτώνοντας τις αντίστοιχες ιστοσελίδες.

Εικόνα 7 – Εκτελώντας μια εφαρμογή παιχνιδιού στο Chrome Os η οποία τρέχει από ιστοτόπο.

1.2 Στόχος Διπλωματικής

Παρατηρώντας αυτές τις τάσεις, θελήσαμε να δημιουργήσουμε μια εφαρμογή, με αντικείμενο που εντάσσεται στον τομέα της αναπαράστασης γραφικών με την βοήθεια υπολογιστών, αλλά όχι σε desktop μορφή. Η διπλωματική εργασία είχε σαν στόχο την δημιουργία «διαδικτυακής εφαρμογής» (web application) αρχιτεκτονικής σχεδίασης και διακόσμησης εσωτερικών χώρων, η οποία επιτρέπει στον χρήστη να δημιουργεί δισδιάστατα σχέδια κατόψεων (floor plans), από τα οποία αυτόματα παράγεται τρισδιάστατη γεωμετρία και δημιουργούνται τρισδιάστατες σκηνές που παρέχουν αλληλεπίδραση. Κατόπιν ο χρήστης μπορεί να περιηγηθεί σε αυτές και να εποπτεύσει τους χώρους τους οποίους δημιούργησε ο ίδιος.

1.3 Σύντομη Περιγραφή

Καταρχήν η εφαρμογή θα υποστηρίζει ένα σύστημα εγγραφής χρηστών, με το οποίο ο κάθε χρήστης θα μπορεί να δημιουργεί προφίλ και να κάνει login στην προσωπική του σελίδα.

Η εφαρμογή παρέχει την δυνατότητα στον χρήστη, δημιουργίας δισδιάστατων (2d) σχεδίων κατόψεων εσωτερικών/εξωτερικών χώρων (floor plans) προσφέροντας έναν αριθμό από εργαλεία σχεδίασης όπως:

- Εργαλεία σχεδίασης τοίχων.
- Εργαλεία οριοθέτησης δωματίων.
- Εργαλεία σχεδίασης δαπέδων και οροφών.
- Εργαλεία δημιουργίας οπών στους τοίχους ώστε να τοποθετηθούν πόρτες και παράθυρα.
- Εργαλεία μετακίνησης και περιστροφής αντικειμένων.

- Εργαλεία διακόσμησης, τα οποία επιτρέπουν την εφαρμογή χρωμάτων ή υφών σε τοίχους, δάπεδα και οροφές.
- Εργαλεία μετρήσεως αποστάσεων και γωνιών στο σχέδιο κλπ.

Κατόπιν, ο χρήστης αντλώντας από την βάση δεδομένων της εφαρμογής, μπορεί να διακοσμήσει τους χώρους/δωμάτια που έχει σχεδιάσει με έναν αριθμό από διάφορα αντικείμενα όπως:

- Έπιπλα
- Ηλεκτρικές Συσκευές
- Διακοσμητικά Αντικείμενα
- Φωτιστικά
- Πόρτες
- Παράθυρα
- Φυτά
- Αντικείμενα εξωτερικών χώρων
- Είδη υγιεινής κλπ.

Αφού ο χρήστης ολοκληρώσει την σχεδίαση της κάτοψης, μπορεί να αποθηκεύσει το σχέδιο για μελλοντική προεπισκόπηση και επεξεργασία. Η εφαρμογή έχει την δυνατότητα να αναλύει την δισδιάστατη κάτοψη και να παράγει (σύμφωνα πάντα με τις παραμέτρους που έχουν οριστεί) μια τρισδιάστατη γεωμετρική απεικόνιση των χώρων που έχει σχεδιάσει ο χρήστης, επιτρέποντας την περιήγηση του μέσα σε αυτούς και την αλληλεπίδραση με διάφορα τρισδιάστατα αντικείμενα που έχουν τοποθετηθεί. Πέραν των δικών του σχεδίων ο χρήστης θα μπορεί να επισκοπήσει σχέδια άλλων χρηστών και να περιηγηθεί στις τρισδιάστατες απεικονίσεις τους. Συνοψίζοντας, οι βασικές λειτουργίες που υλοποιεί η εφαρμογή μας και θα αναλυθούν εκτενέστερα παρακάτω είναι οι εξής:

1. Εγγραφή χρήστη στην ιστοσελίδα της εφαρμογής.
2. Δημιουργία προφίλ του εγγεγραμμένου χρήστη.
3. Δημιουργία δυσδιάστατων σχεδίων κατόψεων, επεξεργασία και αποθήκευση αυτών.
4. Παραγωγή 3d Γεωμετρικών απεικονίσεων των σχεδίων, δυνατότητα περιήγησης μέσα σε αυτά και αλληλεπίδρασης με τα αντικείμενα που περιέχουν.
5. Βάση δεδομένων με αναλυτικές πληροφορίες για τα αντικείμενα που χρησιμοποιεί ο χρήστης (τύποι αντικειμένων, κατασκευαστές, περιγραφές, καταστήματα, τιμές κλπ).
6. Υποστήριξη της κοινότητας των χρηστών της εφαρμογής, με δυνατότητες σχολιασμών σχεδίων και βαθμολόγησης αυτών.

1.4 Πλατφόρμα

Η εφαρμογή εκτελείται εξολοκλήρου μέσα σε περιβάλλον φυλλομετρητή με την μορφή ιστοσελίδας. Κατά την δημιουργία της δόθηκε βάση στο να αξιοποιηθούν και να συνδυαστούν αρμονικά native τεχνολογίες και στάνταρ του σύγχρονου παγκόσμιου ιστού. Για αυτό τον λόγο επιλέχθηκε ένας φυλλομετρητής, σαν σημείο αναφοράς, πάνω στον οποίο εξελίχθηκε η διαδικασία δημιουργίας της εφαρμογής. Ο φυλλομετρητής αυτός είναι ο Firefox στην έκδοση 3 και μετά. Επίσης, είναι πιθανών, η εφαρμογή να λειτουργεί χωρίς προβλήματα και σε άλλους browsers της αγοράς που υποστηρίζουν σύγχρονα standards και τεχνολογίες του ιστού. Σκοπός μας δεν ήταν να εντρυφήσουμε στην επίλυση αυτών των ασυμβατοτήτων, διότι απαιτούν διαδικαστικές διορθώσεις στον κώδικα, οι οποίες δεν έχουν κάποια ιδιαίτερη δυσκολία αλλά ούτε ερευνητική πρόκληση ή σημασία. Παρόλα αυτά, η εφαρμογή μας τρέχει σε αρκετά περιβάλλοντα και διαφορετικά λειτουργικά συστήματα στα οποία έχει παρουσία ο φυλλομετρητής Firefox. (Windows, MacOS, Linux). Η εφαρμογή υλοποιήθηκε σε περιβάλλον Windows και ο web server της λειτουργεί επίσης σε περιβάλλον Windows. Μέρος του debugging έγινε στον Firefox με το εργαλείο Firebug.

1.5 Βασικά Τεχνικά Χαρακτηριστικά

Η εφαρμογή μας περιλαμβάνει ένα συνδυασμό διαφορετικών τεχνολογιών και επιμέρους κομματιών τα οποία συνεργάζονται μεταξύ τους. Καταρχήν, σαν web application, εδρεύει σε διακομιστή ιστού (web-server), από τον οποίο μεταφέρει τις ιστοσελίδες στον φυλλομετρητή του χρήστη (client). Κάποιες διαδικασίες της εφαρμογής εκτελούνται τοπικά, στο μηχάνημα του χρήστη και κάποιες άλλες στον διακομιστή. Υπεύθυνη για την εκτέλεση των εντολών σε περιβάλλον φυλλομετρητή είναι η Javascript. Από την μεριά του εξυπηρετητή, έχουμε την γλώσσα PHP, ώστε να υλοποιούνται αιτήσεις στην βάση δεδομένων, αναζητήσεις στο σύστημα αρχείων του εξυπηρετητή κλπ. Η μεταξύ τους επικοινωνία πραγματοποιείται χρησιμοποιώντας την τεχνολογία AJAX (Asynchronous Javascript And XML) στην οποία θα αναφερθούμε αναλυτικά σε επόμενα κεφάλαια. Οι τεχνολογίες, οι οποίες χρησιμοποιούνται και θα αναλυθούν εις βάθος σε όλο το υπόλοιπο της εργασίας αυτής, είναι οι παρακάτω:

- Ο διακομιστής ιστού (web server), ο οποίος είναι υπεύθυνος για να διανέμει τις ιστοσελίδες της εφαρμογής στον φυλλομετρητή (browser) του χρήστη. Στην συγκεκριμένη εφαρμογή επιλέξαμε τον Apache Http Server.
- Ο εξυπηρετητής βάσεων δεδομένων (database server), στον οποίον θα αποθηκεύονται όλα τα απαραίτητα δεδομένα της εφαρμογής (κατάλογος χρηστών, κατάλογος αντικειμένων κτλ). Για τις ανάγκες της συγκεκριμένης εφαρμογής επιλέξαμε τον MySql Server.

- Την γλώσσα επισήμανσης HTML (Hypertext Markup Language) για την δημιουργία των ιστοσελίδων στις οποίες στηρίζεται η εφαρμογή μας.
- Την γλώσσα CSS (Cascading Styling Sheets) για τον καθορισμό της εμφάνισης των στοιχείων του gui της εφαρμογής μας καθώς και γενικότερα για το layout κάθε σελίδας.
- Την γλώσσα Javascript για το οποιοδήποτε interactivity και προγραμματιστική διαδικασία πρέπει να συμβεί σε περιβάλλον client. Το μεγαλύτερο και βασικότερο μέρος της λογικής της εφαρμογής μας είναι υλοποιημένο στην γλώσσα αυτή.
- Την γλώσσα PHP (Hypertext Preprocessor) για επικοινωνία είτε με τον server, είτε με την βάση δεδομένων, είτε για την δημιουργία sessions, cookies, αποθήκευση, ανάκτηση και διαγραφή αρχείων στον εξυπηρετητή.
- Την γλώσσα επισημάνσεως XML, η οποία χρησιμοποιείται τόσο στην περιγραφή αρκετών από των δομών δεδομένων που περιλαμβάνει η εφαρμογή μας, όσο και για την δημιουργία του πρότυπου αρχείου που αποθηκεύει τα σχέδια και για την ασύγχρονη επικοινωνία με τον διακομιστή.
- Την μεθοδολογία AJAX (Asynchronous Javascript And XML) που δεν αποτελεί τόσο ξεχωριστή τεχνολογία, όσο συνδυασμό τεχνολογιών, ώστε να επιτρέπεται ασύγχρονη επικοινωνία με τον διακομιστή χωρίς να φορτώνεται άλλη σελίδα. Αυτό επιτρέπει την επίτευξη υψηλής και ευέλικτης διαδραστικότητας, σχεδόν αντίστοιχης μιας desktop εφαρμογής.
- Την γλώσσα περιγραφής SVG (Scalable Vector Graphics) για την δημιουργία δυσδιάστατων διανυσματικών γραφικών στην εφαρμογή τα οποία σε συνδυασμό με την Javascript περιέχουν διαδραστικότητα και βοηθούν στο να υλοποιηθεί ο editor σχεδίασης κατόψεων.
- Την γλώσσα περιγραφής X3D για την δημιουργία των τρισδιάστατων σκηνών στις οποίες ο χρήστης θα περιηγείται και θα αλληλεπιδρά..

Αφού παρουσιάστηκαν συνοπτικά τα τεχνικά τμήματα της εφαρμογής ακολουθούν οι λογικές οντότητες, η καθεμία από τις οποίες περιλαμβάνει συνδυασμό των παραπάνω τμημάτων. Η εφαρμογή θα μπορούσε να χωριστεί στις παρακάτω λογικές οντότητες:

- **Σύστημα Διαχείρισης Βάσεως Δεδομένων**
  Περιλαμβάνει την βάση δεδομένων, τον διακομιστή διαχείρισης της, καθώς και όλα τμήματα κώδικα που μας επιτρέπουν να αποθηκεύουμε και να ανακτούμε δεδομένα από αυτήν.
- **Σύστημα Διαχείρισης Αρχείων του διακομιστή**
  Περιλαμβάνει αρχεία κώδικα PHP, τα οποία διαχειρίζονται το σύστημα αρχείων του εξυπηρετητή, επιτρέποντας δημιουργία φακέλων, αποθήκευση, διαγραφή και ανάκτηση αρχείων που δημιουργούνται από τους χρήστες της εφαρμογής.
- **Σύστημα Διαχείρισης Δυσδιάστατων Διανυσματικών Γραφικών**
  Περιλαμβάνει συγκεκριμένες ιστοσελίδες της εφαρμογής μας που υλοποιούν το περιβάλλον δημιουργίας και επεξεργασίας σχεδίων κατόψεων.

- **Σύστημα Διαχείρισης Τρισδιάστατης Γεωμετρίας**
  Περιλαμβάνει διαδικασίες σε Javascript και κλάσεις για δομές δεδομένων, που περιγράφουν low level παραμέτρους τρισδιάστατων μοντέλων όπως vertices, faces, edges, polygons κλπ.
- **Σύστημα μετατροπής των δισδιάστατων σχεδίων σε τρισδιάστατα μοντέλα**
  Περιλαμβάνει διαδικασίες σε γλώσσα Javascript, οι οποίες αναλύουν το δισδιάστατο σχέδιο και παράγουν την αντίστοιχη τρισδιάστατη γεωμετρία.
- **Σύστημα Διαχείρισης Χρωμάτων**
  Περιλαμβάνει διαδικασίες σε γλώσσα Javascript οι οποίες επιτρέπουν την αναπαράσταση και διαχείριση διάφορων χρωματικών μοντέλων.
- **Βοηθητικές Διαδικασίες**
  Περιλαμβάνονται διαδικασίες τριγωνομετρικών υπολογισμών, μετατροπών μονάδων κτλ.
- **Σύστημα Ασύγχρονης Επικοινωνίας με τον εξυπηρετητή ιστού**
  Περιλαμβάνει συνδυασμό διαδικασιών σε Javascript και PHP οι οποίες ανταλλάσουν δεδομένα (ασύγχρονα), σε μορφή XML.
- **Σύστημα Διαχείρισης Του γραφικού περιβάλλοντος χρήστη.**
  Περιλαμβάνει τις σελίδες HTML οι οποίες σε συνδυασμό με διαδικασίες events σε Javascript παρέχουν διαδραστικότητα και αποτελούν την γραφική διεπαφή (User Interface) της εφαρμογής.

1.6 Δομή της εργασίας

Το τρέχων εισαγωγικό κεφάλαιο ακολουθούν 6 ακόμη κεφάλαια, καθένα από τα οποία περιγράφεται συνοπτικά παρακάτω:

- Στο δεύτερο κεφάλαιο αναλύεται η διαδικασία της έρευνας, η οποία πραγματοποιήθηκε σχετικά με την διπλωματική εργασία. Παρουσιάζονται κάποιοι από τους διαθέσιμους διακομιστές ιστού και βάσεων δεδομένων. Επίσης εξετάζονται οι διαθέσιμες τεχνολογίες εκτέλεσης κώδικα, τόσο σε περιβάλλον διακομιστή (PHP,ASP,Java Server Pages) όσο και σε περιβάλλον φυλλομετρητή (Javascript, ActionScript, Java Applets κτλ.). Τέλος εξετάζονται οι διαθέσιμες τεχνολογίες αναπαράστασης δισδιάστατων αλλά και τρισδιάστατων γραφικών σε περιβάλλον φυλλομετρητή (SVG, vml, flash, java 2d, X3D, unity3d player, Silverlight, Google O3D, Webgl). Αναφέρονται τα πλεονεκτήματα και τα μειονεκτήματα τις καθεμίας τεχνολογίας, καθώς και οι λόγοι για τους οποίους επιλέχθηκαν τελικά αυτές που ενσωματώθηκαν στην τελική μορφή της εφαρμογής.

- Στο τρίτο κεφάλαιο, γίνεται αρχικά μια περιγραφή του περιβάλλοντος του διακομιστή ιστού (web server), στο οποίο εδρεύει η εφαρμογή, αλλά και του περιβάλλοντος του φυλλομετρητή, στο οποίο εκτελείται ένα μεγάλο μέρος της. Επίσης, περιγράφεται η αρχιτεκτονική της κάθε τεχνολογίας που χρησιμοποιήθηκε για την υλοποίηση, ο σκοπός της και τα σημεία της εφαρμογής στα οποία ενσωματώνεται.

- Στο τέταρτο κεφάλαιο περιγράφεται η αρχιτεκτονική της εφαρμογής καθώς επίσης και τα στάδια μελέτης και φιλοσοφίας τα οποία οδήγησαν στην τελική μορφή της. Αναλύονται τα επιμέρους τμήματα και περιγράφονται η αναγκαιότητά τους για την πλήρη λειτουργικότητα του συνόλου καθώς και η μεταξύ τους αλληλεπίδραση.

- Στο πέμπτο κεφάλαιο αναλύονται με λεπτομέρεια οι τρόποι με τους οποίους υλοποιήθηκε η συγκεκριμένη αρχιτεκτονική που επιλέξαμε. Παρουσιάζονται κομμάτια διαδικασιών, αλγορίθμων και κώδικα από καίρια κομμάτια της εφαρμογής καθώς επίσης αναλύονται και τα προβλήματα τα οποία επιλύουν. Τέλος γίνεται μια τεχνική αποτίμηση όλων των μονάδων που συμβάλλουν στην υλοποίηση και λειτουργία της εφαρμογής.

- Το έκτο κεφάλαιο αφιερώνεται στα αποτελέσματα της αξιολόγηση της εφαρμογής από διάφορους χρήστες, στην ανακεφαλαίωση της διπλωματικής εργασίας, στα τελικά συμπεράσματα και στις μελλοντικές επεκτάσεις.

ΚΕΦΑΛΑΙΟ 2
ΕΠΙΣΚΟΠΗΣΗ ΣΧΕΤΙΚΗΣ ΕΡΕΥΝΑΣ

2.1 Εισαγωγή

Στο κεφάλαιο αυτό αναφέρονται οι τρόποι και οι μεθοδολογίες με τους οποίους δομείται η ανάπτυξη μιας διαδικτυακής εφαρμογής (web application). Επίσης παρουσιάζονται οι απαραίτητες τεχνολογίες για την υλοποίησή της. Επιπροσθέτως, παρουσιάζεται και μια σειρά από τεχνολογίες (SVG, Flash, VML, Silverlight, Java 2d, X3D, Webgl, Google O3D κτλ) για την παρουσίαση διαδραστικών δισδιάστατων και τρισδιάστατων γραφικών, ως ενσωματωμένα αντικείμενα ιστοσελίδων, μιας και κάτι τέτοιο είναι απαραίτητο για τις λειτουργίες που θα πρέπει να υποστηρίξουμε στο πρόγραμμά μας.

2.2 Γενικές απαιτήσεις Διαδικτυακών Εφαρμογών

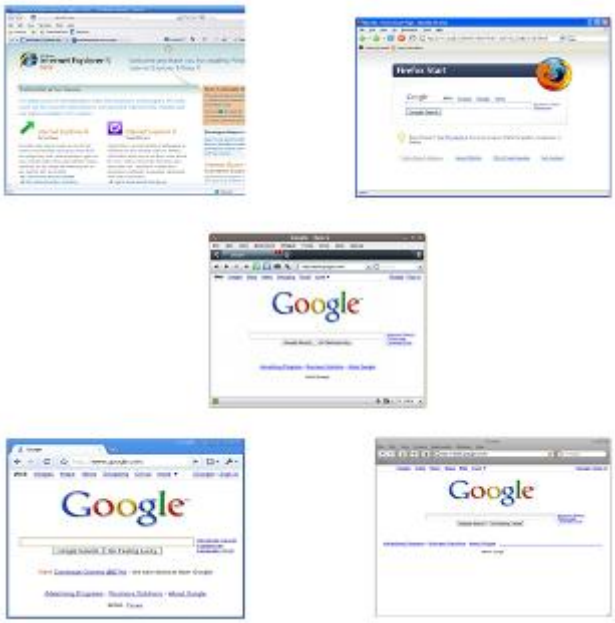
Όπως αναφέραμε και στο προηγούμενο κεφάλαιο, ο όρος διαδικτυακή εφαρμογή αναφέρεται στις εφαρμογές εκείνες, οι οποίες είναι προσβάσιμες μέσω διαδικτύου και χρησιμοποιούν μια γλώσσα προγραμματισμού που υποστηρίζεται από φυλλομετρητές (όπως π.χ. η Javascript), σε συνδυασμό με μια γλώσσα επισήμανσης για την παρουσίαση της γραφικής διεπαφής τους (η οποία επίσης υποστηρίζεται από τους φυλλομετρητές, όπως η HTML). Βασίζονται σε έναν συνηθισμένο φυλλομετρητή ιστοσελίδων (browser) ώστε να τις θέτει εκτελέσιμες. Για την αποθήκευση δεδομένων θα πρέπει να υπάρχει επικοινωνία με έναν διακομιστή ιστού (web server). Η επικοινωνία μεταξύ client και server, υλοποιείται με μια scripting γλώσσα προγραμματισμού (ASP,PHP κτλ.). Ο web server είναι, ούτως η άλλως, απαραίτητος για να φιλοξενεί τα αρχεία της εφαρμογής μας αν θέλουμε αυτή να είναι διαθέσιμη στον παγκόσμιο ιστό. Οι web applications ανάλογα με τις λειτουργίες που υποστηρίζουν χωρίζονται συνήθως σε 3 βασικές βαθμίδες:

- Βαθμίδα Παρουσίασης
- Βαθμίδα Εκτέλεσης
- Βαθμίδα Αποθήκευσης

Ο φυλλομετρητής αποτελεί την βαθμίδα παρουσίασης, ένας διακομιστής ιστού, που χρησιμοποιεί μια από τις τεχνολογίες υποστήριξης δυναμικού web περιεχομένου, αποτελεί την βαθμίδα εκτέλεσης και η βάση δεδομένων αποτελεί την βαθμίδα αποθήκευσης.

2.3 Βαθμίδα Παρουσίασης : Ο φυλλομετρητής (Web browser)

Στην αγορά, υπάρχει ένας τεράστιος αριθμός φυλλομετρητών, από τους οποίους οι περισσότεροι διατίθενται δωρεάν και άλλοι επί πληρωμή. Από τους πιο διαδεδομένους είναι οι Internet Explorer, Mozilla Firefox, Google Chrome, Opera και Apple Safari (Εικόνα 8).



Εικόνα 8 - Screenshots από του 5 δημοφιλέστερους browsers της αγοράς. Από αριστερά προς τα δεξιά: Internet Explorer 9, Mozilla Firefox 3.6, Opera, Google Chrome και Apple Safari.

Η επιλογή του φυλλομετρητή φυσικά δεν εξαρτάται από εμάς, αλλά από τον τελικό χρήστη που θα χρησιμοποιήσει την εφαρμογή μας. Δυστυχώς δεν υπάρχει εγγύηση ότι η εφαρμογή θα εκτελείται το ίδιο και το γραφικό της περιβάλλον θα παρουσιάζεται με τον ίδιο τρόπο σε κάθε φυλλομετρητή. Αυτό οφείλεται στο γεγονός ότι υπάρχουν ασυμφωνίες στην υποστήριξη διαδικτυακών τεχνολογιών (όπως π.χ. στον τρόπο με τον οποίο εφαρμόζονται οι κανόνες παρουσίασης στα Cascading Style Sheets, ή και στον τρόπο που ερμηνεύονται μερικές φορές οι εντολές της Javascript) μεταξύ των browsers. Για αυτό τον λόγο γίνεται προσπάθεια εδώ και αρκετά χρόνια να προωθούνται διάφορα στάνταρ όσον αφορά τις web τεχνολογίες, αλλά δυστυχώς οι κατασκευαστές των φυλλομετρητών δεν τα υποστηρίζουν στον ίδιο βαθμό. Έτσι, από την πλευρά των προγραμματιστών και web developers, γίνεται μια τεράστια προσπάθεια ώστε οι ιστοτόποι να υποστηρίζουν όσο το δυνατόν μεγαλύτερο αριθμό φυλλομετρητών, γράφοντας πολλαπλά εναλλακτικά κομμάτια κώδικα τα οποία διορθώνουν αυτά τα προβλήματα ασυμβατότητας. Στα πλαίσια όμως αυτής της διπλωματικής επικεντρωθήκαμε σε έναν μόνο browser, ώστε να αποφύγουμε να ασχοληθούμε με τα προβλήματα ασυμβατότητας, η υλοποίηση των οποίων είναι χρονοβόρα και δεν έχει καμία ερευνητική αξία. Επιλέχθηκε ο Mozilla Firefox, ως περιβάλλον για τις δοκιμές εκτέλεσης της εφαρμογής μας και αυτό έγινε διότι αποτελεί έναν φυλλομετρητή με μεγάλη απήχηση στο κοινό, είναι open-source project, υποστηρίζει σωστά έναν πάρα πολύ μεγάλο βαθμό από web-standards και

τεχνολογίες, είναι αρκετά γρήγορος στην εκτέλεση της Javascript (αν και ο Chrome τον ξεπερνάει σε ταχύτητα) και προσφέρει μια μεγάλη ποικιλία εργαλείων για προγραμματιστές (για live ανάλυση των αντικειμένων μιας ιστοσελίδας, για debugging και profiling κώδικα Javascript κτλ). Επίσης προσφέρεται και σε κάποιες δοκιμαστικές εκδόσεις (nightly builds), οι οποίες περιέχουν υποστήριξη για κάποιες νέες τεχνολογίες που βρίσκονται ακόμα σε πειραματικό στάδιο. Η αξιοποίηση τους όμως, έστω και τώρα, έχει μεγάλο ενδιαφέρον και προσφέρει μια «ματιά» στο κοντινό μέλλον του web.

2.4 Βαθμίδα Εκτέλεσης : Ο διακομιστής ιστού (Web Server)

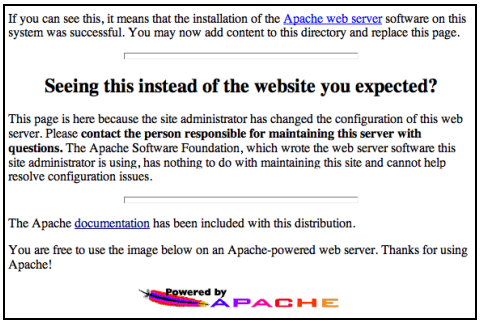
Αντίθετα με τους φυλλομετρητές, στο τοπίο των διακομιστών, οι επιλογές είναι λιγότερες. Κατά την έρευνά μας, ξεχωρίσαμε τους εξής:

2.4.1 Microsoft Internet Information Services

Εικόνα 9 - Αρχική σελίδα του διακομιστή IIS της Microsoft.

Αποτελεί την λύση τις Microsoft στον τομέα των web-servers. Υποστηρίζει αρκετές web τεχνολογίες και ιδιαίτερα εκείνες που αποτελούν δημιουργίες της Microsoft, όπως Active Server Pages κλπ. Κατέχει το 24.47% μερίδιο της αγοράς. Σε προηγούμενες εκδόσεις υπήρχαν κάποια θέματα σταθερότητας. Ένα πλεονέκτημα είναι ότι συνεργάζεται άψογα με το λειτουργικό των Windows (σε αντίθεση με άλλους servers). Στα μειονεκτήματα του συγκαταλέγονται το γεγονός ότι δεν είναι δωρεάν, ότι είναι κλειστού κώδικα και ότι τρέχει μόνο σε περιβάλλον Windows (Εικόνα 9).

2.4.2 Apache HTTP Server



Εικόνα 10 – Η σελίδα επιβεβαίωσης της σωστής εγκατάστασης και λειτουργίας του apache web server

Ένας ιστορικής σημασίας Web Server ο οποίος συνέβαλε στην ανάπτυξη του αρχικού παγκόσμιου ιστού. Αποτελεί τον κυρίαρχο ανάμεσα στους διακομιστές web με μερίδιο αγοράς 51%. Διατίθεται δωρεάν, ως project ανοιχτού κώδικα από το Apache Software Foundation και υποστηρίζει έναν αριθμό από λειτουργικά (όπως Windows, Linux, Mac Os) και διακρίνεται για την σταθερότητα, την ταχύτητα, και την ασφάλειά του. Επίσης είναι ευέλικτος και μπορεί να επεκταθεί με πολλά modules. Υποστηρίζει εγγενώς αρκετές γλώσσες προγραμματισμού όπως PHP. Επίσης, έχει μια τεράστια κοινότητα χρηστών, οι οποίοι τον υποστηρίζουν και παρέχουν βοήθεια μέσα από forums, tutorials κλπ (Εικόνα 10).

2.4.3 Αιτιολόγηση επιλογής του Apache Http Server

Επιλέξαμε τον **Apache Http Server,** πρώτα από όλα διότι διακρίνεται για την σταθερότητα και την ποιότητα του (χρησιμοποιείται για να φιλοξενεί sites κολοσσών στο internet) και ταυτόχρονα είναι δωρεάν. Το γεγονός ότι μπορεί να εγκατασταθεί και να λειτουργήσει και σε άλλα λειτουργικά συστήματα, πέραν των Windows, προσθέτει μια ευελιξία στην εφαρμογή μας όσον αφορά το portability. Επίσης μπορεί εύκολα να εγκατασταθεί, τοπικά στον υπολογιστή, ώστε ο προγραμματιστής να ελέγξει την ορθή εκτέλεση της διαδικτυακή εφαρμογής του, πριν την ανεβάσει στον παγκόσμιο ιστό.

2.5 Βαθμίδα Αποθήκευσης : Σύστημα βάσης δεδομένων

Ένας διακομιστής βάσεων δεδομένων είναι απαραίτητος όταν η εφαρμογή μας απαιτεί την αποθήκευση δεδομένων που δημιουργούνται από τον χρήστη. Τα συστήματα βάσεων δεδομένων έρχονται με την μορφή εξυπηρετητών (Database Servers) οι οποίοι συνεργάζονται με τον διακομιστή ιστού (web server). Συνήθως τα συστήματα αυτά, υποστηρίζουν ερωτήματα (queries) προς την βάση με την μορφή scripts σε γλώσσα SQL. Τα συστήματα τα οποία διερευνήσαμε αναφέρονται παρακάτω:

2.5.1 Microsoft SQL Server

Συνεργάζεται αρμονικά με τον Web Server της Microsoft που αναφέραμε παραπάνω. Υποστηρίζει ερωτήματα σε γλώσσα SQL. Η τιμή του είναι υψηλή και απευθύνεται σε περιπτώσεις επαγγελματικής χρήσης. Υποστηρίζει μόνο περιβάλλον Windows και είναι κλειστού κώδικα.

2.5.2 Oracle Server

Ο Oracle Server αποτελεί πρόταση της Oracle, η οποία είναι φημισμένη εταιρία που ασχολείται αποκλειστικά με συστήματα διαχείρισης βάσεων δεδομένων. Η τιμή είναι υψηλή και απευθύνεται σε εμπορικές λύσεις.

2.5.3 MySQL

Η MySQL αποτελεί ένα σύστημα διαχείρισης βάσεων δεδομένων που λειτουργεί ως διακομιστής (server), παρέχοντας πρόσβαση πολλών χρηστών σε έναν μεγάλο αριθμό από βάσεις. Αποτελεί δωρεάν, ανοιχτού κώδικα, λογισμικό το οποίο διακρίνεται έναντι των εμπορικών αντιπάλων του για την σταθερότητα και της δυνατότητές του. Χρησιμοποιείται σε μεγάλης κλίμακας web-projects όπως η Wikipedia, το Facebook κλπ. Υποστηρίζει έναν τεράστιο αριθμό από λειτουργικά συστήματα. Επίσης σχεδόν κάθε γνωστή γλώσσα προγραμματισμού, είτε για desktop εφαρμογές είτε για web applications, παρέχει βιβλιοθήκες με bindings για επικοινωνία με τον MySql server.

2.5.4 Αιτιολόγηση επιλογής του MySql Server

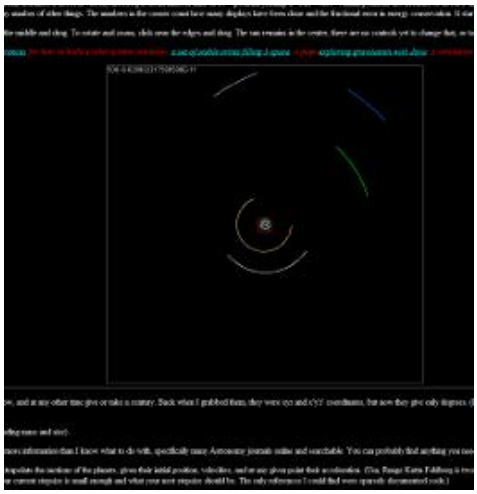
Επιλέξαμε τον **MySQL Server** για τα πλεονεκτήματα που προσφέρει όσο αφορά την τιμή, τις δυνατότητες, την ευρεία υποστήριξή του, και τον ανοιχτού κώδικα χαρακτήρα. Το τελευταίο, αποτελεί ένα σημαντικό κριτήριο στην δημιουργία της εφαρμογής μας, δηλαδή να χρησιμοποιηθούν όσον τον δυνατόν περισσότερα ανοιχτού κώδικα (open-source) εργαλεία και τεχνολογίες.

2.6 Τεχνολογίες ανάπτυξης διαδικτυακών εφαρμογών

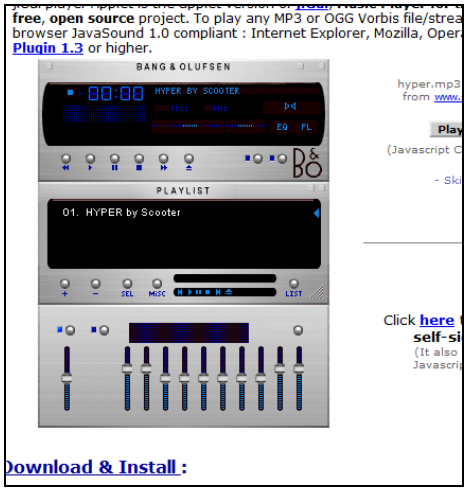
Επειδή η εφαρμογή μας επιτρέπει την δημιουργία περιεχομένου και μάλιστα περίπλοκου (δυσδιάστατα γραφικά και τρισδιάστατες αναπαραστάσεις), η χρησιμοποίηση μόνο στατικών σελίδων HTML δεν αρκεί. Παρακάτω αναφέρονται κάποιες τεχνολογίες, με τις οποίες θα μπορούσε να αναπτυχθεί η εφαρμογή μέσα σε περιβάλλον φυλλομετρητή.

2.6.1 Java Applet

Πριν εξελιχθούν οι φυλλομετρητές, ώστε να υποστηρίζουν τις σύγχρονες τεχνολογίες που χαρακτηρίζουν το web 2.0, ένας από τους παλιότερους τρόπους για να υλοποιηθούν περίπλοκες εφαρμογές μέσα σε ιστοσελίδες, ήταν η χρήση Java Applets. Τα applets αυτά περιέχουν εκτελέσιμο κώδικα Java σε μορφή Java Byte Code. Για να εκτελεστούν θα πρέπει να είναι εγκατεστημένη στο σύστημα του χρήστη, η Java Virtual Machine. Τα applets ενσωματώνονται ως συγκεκριμένα πλαίσια, μέσα στην σελίδα και είναι διαχωρισμένα από τον υπόλοιπο κορμό της (εικόνες 11 και 12).



Εικόνα 11 - Java Applet το οποίο αναπαριστά τις τροχιές των πλανητών του ηλιακού μας συστήματος βρίσκεται ενσωματωμένο μέσα σε ιστοσελίδα αστρονομίας.



Εικόνα 12 - Java applet που υλοποιεί έναν ολοκληρωμένο mp3 player.

Πλεονεκτήματα:

- Η εφαρμογή σε μορφή Applet εκτελείται με τον ίδιο τρόπο, σε διαφορετικά λειτουργικά συστήματα, σε διαφορετικούς φυλλομετρητές, αρκεί να υπάρχει εγκατεστημένο το Java Runtime Environment (JRE).
- Τα Applets αποθηκεύονται στην cache των περισσότερων φυλλομετρητών, οπότε συμβάλλουν στο γρήγορο φόρτωμα της εφαρμογής, όταν επιστρέψουμε στον ιστοτόπο της.
- Παρέχουν την ευελιξία και την δύναμη της γλώσσας προγραμματισμού Java και επιτρέπουν την δημιουργία εφαρμογών επιπέδου desktop, χρησιμοποιώντας έναν τεράστιο αριθμό βιβλιοθηκών (όπως π.χ. εφαρμογές 3d χρησιμοποιώντας την βιβλιοθήκη java3d).
- Η εκτέλεσή των Java Applets είναι γρήγορη, σχεδόν σαν να ήταν desktop εφαρμογές.
- Παρέχουν ασφάλεια καθότι αν ένα applet είναι untrusted δεν έχει πρόσβαση στο σύστημα αρχείων του συστήματος – πελάτη.

Μειονεκτήματα:

- Απαιτούν την ύπαρξη Java plug-in, κάτι το οποίο είναι ενάντια στην φιλοσοφία σχεδίασης της εφαρμογής μας. Θέλουμε να αποφύγουμε την χρήση τέτοιων εξαρτήσεων, ώστε η λειτουργία της εφαρμογής να εξαρτάται όσον το δυνατόν περισσότερο από τον φυλλομετρητή και όχι από εξωτερικούς παράγοντες όπως πρόσθετο εγκατεστημένο λογισμικό και plug-ins.
- Μερικά Applets απαιτούν συγκεκριμένες εκδόσεις της Java για να τρέξουν σωστά.

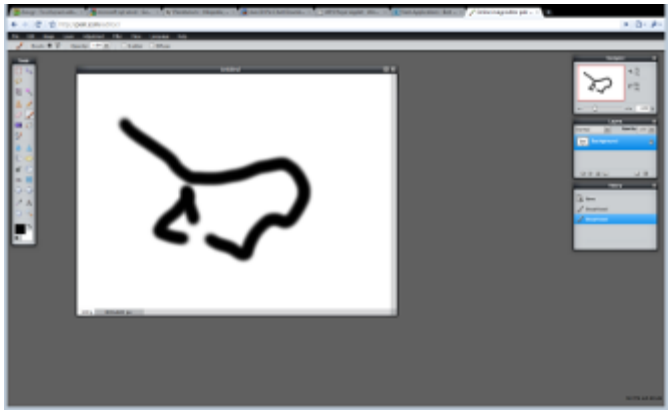
2.6.2 Microsoft Active-X

Παρόμοια τεχνολογία αποτελεί από την πλευρά της Microsoft, τα ActiveX controls, τα οποία επιτρέπουν την εκτέλεση εφαρμογών μέσα σε συγκεκριμένα πλαίσια της σελίδας. Οι εφαρμογές αυτές θα πρέπει να είναι υλοποιημένες σε γλώσσες C++, Borland Delphi ή Microsoft Visual Basic. Υποστηρίζονται μόνο από τον Internet Explorer.

2.6.3 Flash

Το flash δεν χρειάζεται ιδιαίτερες συστάσεις, καθώς χρησιμοποιείται κατά κόρον σε sites, για να προσθέσει πολυμεσικές λειτουργίες όπως κινούμενα γραφικά, εικόνα, βίντεο ήχο κλπ. Αρχικά είχε μόνο διακοσμητικό ρόλο στις ιστοσελίδες, αλλά με το πέρασμα των χρόνων έχει εξελιχθεί σε μια πολύ δυνατή πλατφόρμα ανάπτυξης εφαρμογών (εικόνες 13, 14 και 15).

Εικόνα 13 - Παιχνίδι υλοποιημένο σε flash.



Εικόνα 14 – Εφαρμογή επεξεργασίας εικόνας μέσα σε περιβάλλον browser υλοποιημένη εξ ολοκλήρου σε flash.

Εικόνα 15 - Σχεδόν όλα τα sites παρουσίασης video όπως το youtube χρησιμοποιούν τεχνολογία flash για την αναπαραγωγή του.

Πλεονεκτήματα:

- Παρέχει δυνατότητες ισχυρής και ταχύτατης επεξεργασίας, τόσο διανυσματικών όσο και raster-based γραφικών (γραφικά βασισμένα σε bitmaps). Επιτρέπει την δημιουργία animation με κείμενο, εικόνες και σχέδια.
- Υποστηρίζει αμφίπλευρο streaming ήχου και εικόνας. Για αυτό τον λόγο χρησιμοποιείται κατά κόρον στο σύγχρονο web, σε ιστοσελίδες ιντερνετικής τηλεόρασης, ραδιοφώνου κλπ (εικόνα 2.9).
- Επιτρέπει την διαχείριση events εισόδου από το πληκτρολόγιο, το ποντίκι, το μικρόφωνο και την web camera.
- Το flash περιέχει μια πανίσχυρη αντικειμενοστραφή γλώσσα προγραμματισμού, την Action Script, βασισμένη στο πρότυπο Ecmascript. Περιλαμβάνει low level τύπους δεδομένων, δημιουργία συναρτήσεων, δημιουργία κλάσεων και αντικειμένων, κληρονομικότητα, packages, namespaces και regular expressions, action listeners, event handling κτλ. Επίσης περιλαμβάνει βιβλιοθήκες για δημιουργία αντικειμένων φορμών όπως textboxes, sliders, tabs κτλ.
- Το Flash περιέχει περιορισμένη υποστήριξη για επιταχυνόμενα γραφικά (Opengl και DirectX).

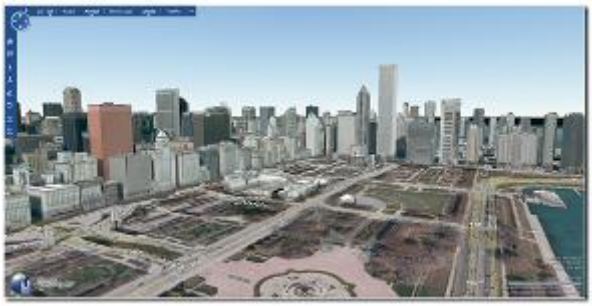
Μειονεκτήματα:

- Απαιτεί την ύπαρξη plugin.
- Ορισμένες φορές, η εκτέλεση των εφαρμογών, δεσμεύει αρκετούς πόρους από το σύστημα σε σχέση με άλλες τεχνολογίες.
- Δεν υπάρχει αφομοίωση μέσα στο σώμα της HTML σελίδας, καθώς οι εφαρμογές του flash όπως και τα applets της java, βρίσκονται μέσα σε τετράγωνα πλαίσια, τα οποία διαχωρίζονται από τα υπόλοιπα στοιχεία της σελίδας.

- Η επικοινωνία με τον διακομιστή, μπορεί να γίνει μέσω PHP, αλλά πιο δύσκολα από ότι π.χ. μεταξύ Javascript και PHP.
- Η υποστήριξη τρισδιάστατων γραφικών, είναι σε πρώιμο στάδιο και γίνεται με software-rendering λύσεις οι οποίες δεσμεύουν αρκετούς πόρους για να εκτελεστούν.
- Υπάρχει περίπτωση να αποκλειστεί από εργαλεία μπλοκαρίσματος ιντερνετικών διαφημίσεων.

2.6.4 Microsoft Silverlight

Το ακριβώς ανάλογο της τεχνολογίας Flash από την πλευρά της Microsoft. Παρουσιάστηκε σχετικά αργά (2007) και επιτρέπει την διαχείριση διανυσματικών γραφικών , streaming ήχου και εικόνας καθώς και interactivity μέσω της γλώσσας XAML. Απαιτεί την ύπαρξη plug-in και έχει πολύ μικρότερη δημοτικότητα σε χρήστες από ότι το flash το οποίο υπάρχει από το 1995 (Εικόνα 16).



Εικόνα 16 - Η εφαρμογή bing maps 3d υλοποιημένη πλήρως σε Silverlight.

2.6.5 Javascript

Η γλώσσα προγραμματισμού Javascript, αποτελεί μια υλοποίηση της Ecmascript και υλοποιείται ως αναπόσπαστο κομμάτι του εκάστοτε browser (ο οποίος στηρίζεται σε μια μηχανή εκτέλεσης της Javascript). Παρέχει δυνατότητα δημιουργίας προγραμματιστικών διαδικασιών μέσα σε μια ιστοσελίδα, δίνοντας πρόσβαση στα αντικείμενα αυτής, μέσω του Document Object Model. Επίσης δίνει προγραμματιστική πρόσβαση σε native αντικείμενα του φυλλομετρητή τα οποία μπορούμε να αλλάξουμε μέσω κώδικα (π.χ. το μέγεθος του παραθύρου του φυλλομετρητή, τα χρώματα των scrollbars κλπ). Επειδή αρχικά, η υποστήριξη της Javascript ήταν μηδαμινή και γενικά η χρήση της περιοριζότανε, επί το πλείστον, στην δημιουργία κάποιων μικρών animation σε σελίδες και σε απλές επικυρώσεις

φορμών, δημιουργήθηκε η λανθασμένη άποψη (η οποία δυστυχώς εκφράζεται και σήμερα από αρκετούς) ότι η γλώσσα αυτή έχει περιορισμένες δυνατότητες. Κάτι τέτοιο όμως δεν ισχύει. Η Javascript περιλαμβάνει σύνταξη σε στυλ C, με κλασσικούς τύπους δεδομένων που συναντάμε σε κανονικές γλώσσες προγραμματισμού, δυνατότητα δημιουργίας μεθόδων και συναρτήσεων, ευέλικτης δημιουργίας κλάσεων και αντικειμένων, μηχανισμούς κληρονομικότητας, ασύγχρονη επικοινωνία με τον διακομιστή, μηχανισμούς event handling, πρόσβαση σε αντικείμενα των html σελίδων μέσω του Document Object Model, πρόσβαση γενικότερα σε αντικείμενα XML γλωσσών όπως π.χ. SVG, MathML. Η Javascript έχει πολλές native μεθόδους για την διαχείριση αντικειμένων, καθώς τα πάντα στην γλώσσα αυτή είναι αντικείμενα (objects). Η Javascript περιλαμβάνει την δομή των κλασσικών γλωσσών προγραμματισμού όπως η C, με block statements, δομές επαναλήψεων, δομές if κτλ. Υποστηρίζει δυναμικό ορισμό τύπων μεταβλητών. Δηλ. μια μεταβλητή x δεν δεσμεύεται από ένα συγκεκριμένο τύπο κατά την δήλωσή της, αλλά μπορεί να αλλάξει κατά την διάρκεια της εκτέλεσης (πχ από ακέραιος σε συμβολοσειρά). Η Javascript περιλαμβάνει και χαρακτηριστικά που «λείπουν» από αντίστοιχες γλώσσες ανάπτυξης desktop εφαρμογών. Οι συναρτήσεις στην Javascript είναι επίσης αντικείμενα με ιδιότητες. Οι συναρτήσεις μπορούν να χρησιμοποιηθούν με το keyword new ως object constructors. Για να υλοποιηθεί η κληρονομικότητα χρησιμοποιούνται τα prototypes για τα οποία θα μιλήσουμε αναλυτικά στο επόμενο κεφάλαιο. Ένα μειονέκτημα τις γλώσσας, είναι ότι λόγω τις δυναμικής της μορφής η εκτέλεση υστερεί σε ταχύτητα. Αυτό το χαρακτηριστικό όμως συνεχώς βελτιώνεται, χάρη στην, συνεχώς, αυξημένη ταχύτητα των μηχανών διερμήνευσης της γλώσσας. Ένα πάρα πολύ σημαντικό χαρακτηριστικό που βοήθησε στην δημιουργία πλούσιων ιντερνετικών εφαρμογών ήταν η δυνατότητα ασύγχρονης επικοινωνίας με τον εξυπηρετητή, χωρίς να φορτωθεί ξανά η σελίδα (κάτι που θυμίζει την ρευστότητα στην διαδραστικότητα μιας desktop εφαρμογής). Η Javascript μπορεί να αλλάξει παραμέτρους σε CSS αρχεία, τα οποία είναι υπεύθυνα για την διάταξη και την σωστή εμφάνιση των αντικειμένων HTML. Για όλα τα παραπάνω πλεονεκτήματα, επιλέξαμε την **Javascript** για την ανάπτυξη του τμήματος της εφαρμογής που εκτελείται στο περιβάλλον φυλλομετρητή.

2.7 Επικοινωνία με τον διακομιστή

Επειδή η διαδικτυακή μας εφαρμογή, υποστηρίζει λειτουργίες δημιουργίας περιεχομένου από τους χρήστες, αυτό σημαίνει ότι θα έχουμε ένα σύνολο δεδομένων τα οποία θα πρέπει να αποθηκεύονται κεντρικά στον διακομιστή. Αυτό απαιτεί την ύπαρξη μιας γλώσσας προγραμματισμού η οποία θα εκτελείται στο περιβάλλον του web server και θα επιτρέπει την επικοινωνία με το σύστημα της βάσης δεδομένων και με το σύστημα αρχείων του διακομιστή ιστού. Στην έρευνά μας εξετάσαμε αρκετές γλώσσες προγραμματισμού, οι οποίες χρησιμοποιούνται για την δημιουργία δυναμικών ιστοσελίδων και εκτελούνται στον server.

2.7.1 PHP

Η γλώσσα PHP αποτελεί μια από τις πιο διαδεδομένες λύσεις στο διαδίκτυο για την κατασκευή δυναμικών ιστοσελίδων. Εκτελείται σε περιβάλλον εξυπηρετητή σε αντίθεση π.χ. με την Javascript η οποία εκτελείται στον φυλλομετρητή του χρήστη. Μια σελίδα PHP στην ουσία είναι μια σελίδα HTML, η οποία περιέχει ενσωματωμένο κώδικα εντολών PHP. Ο διακομιστής την αναγνωρίζει, διαβάζοντας την κατάληξη του αρχείου (η οποία θα πρέπει να είναι *.PHP, *.phtml) και προ-επεξεργάζεται τα blocks εντολών PHP, παράγοντας δυναμικά το περιεχόμενο της σελίδας. Η γλώσσα προσφέρει έναν τεράστιο αριθμό από έτοιμες βιβλιοθήκες, για επικοινωνία με βάσεις δεδομένων, ανάλυση αρχείων xml, επεξεργασία εικόνων, δημιουργία αρχείων pdf, επικοινωνία με mail-servers κλπ. Ο διακομιστής apache που αναφέρθηκε παραπάνω επιτρέπει αυτόματα την εκτέλεση κώδικα PHP χρησιμοποιώντας ειδικά modules που περιλαμβάνουν και τον interpreter της γλώσσας. Η PHP υποστηρίζει οντοκεντρικό προγραμματισμό (κληρονομικότητα, δημιουργία κλάσεων). Επίσης περιλαμβάνει συναρτήσεις για εγγραφή/ανάγνωση αρχείων στους φακέλους του server. Χρησιμοποιείται σε πολύ μεγάλες κλίμακας ιστοσελίδες, όπως το Facebook, το Amazon.com κλπ.

2.7.2 Active Server Pages (ASP)

Οι Active Server Pages, ή αλλιώς ASP, αποτελούν την προσπάθεια της Microsoft για την δημιουργία server-side περιβάλλοντος ανάπτυξης δυναμικών ιστοσελίδων. Αποτελούν αναπόσπαστο κομμάτι του διακομιστή Microsoft Internet Information Services. Ο προγραμματισμός γίνεται σε γλώσσα Vbscript ή Jscript (Η υλοποίηση της Microsoft του προτύπου Ecmascript). Η μηχανή μεταγλώττισης των Active Server Pages υποστηρίζει το πρότυπο Component Object Model της Microsoft, το οποίο επιτρέπει πρόσβαση σε ήδη μεταγλωττισμένες βιβλιοθήκες, όπως αρχεία dll. Η επόμενη εξέλιξη της τεχνολογίας ASP είναι η ASP.NET η οποία χρησιμοποιεί την πλατφόρμα .Net της Microsoft. Το ASP.NET περιλαμβάνει κώδικα, ο οποίος είναι ήδη μεταγλωττισμένος και έτσι αποφεύγεται η διερμήνευσή του κατά την εκτέλεση, κάτι το οποίο προσφέρει μεγάλη ταχύτητα στο φόρτωμα των σελίδων. Η εκτέλεσή τους απαιτεί την ύπαρξη περιβάλλοντος Net Framework και Windows καθώς επίσης και τον εξυπηρετητή της Microsoft.

2.7.3 Java Server Pages (JSP)

Οι Java Server Pages αποτελούν μια τεχνολογία δημιουργίας δυναμικών ιστοσελίδων βασισμένη στην γλώσσα προγραμματισμού JAVA και χρησιμοποιώντας αρχεία HTML και XML. Εμφανίστηκαν το 1999 ως απάντηση της εταιρίας SUN στις τεχνολογίες PHP και ASP. Η τεχνολογία JSP επιτρέπει την εκτέλεση κώδικα JAVA ο οποίος ενσωματώνεται μέσα στο σώμα σελίδων HTML. Αφού μεταγλωττιστούν και εκτελεστούν τα block του κώδικα, τα αποτελέσματα συνδυάζονται με τα στοιχεία

html που τα περιβάλλουν. Έτσι παράγεται η τελική μορφή της σελίδας html, η οποία παρουσιάζεται στο φυλλομετρητή του χρήστη. Η εκτέλεση του κώδικα java και η αναφορά στις βιβλιοθήκες της γλώσσας πραγματοποιείται από την ύπαρξη μιας JAVA Virtual Machine στον server. Επειδή η Java αποτελεί compiled γλώσσα προγραμματισμού και όχι scripting γλώσσα, ο κώδικας των σελίδων JSP μεταγλωττίζεται σε java byte code πριν εκτελεστεί. Έτσι επιτυγχάνεται μεγάλη ταχύτητα φόρτωσης και εκτέλεσης των σελίδων του ιστοτόπου.

2.7.4 Αιτιολόγηση επιλογής της PHP ως γλώσσα προγραμματισμού στο περιβάλλον του εξυπηρετητή

Από τις τεχνολογίες server-side scripting που ερευνήθηκαν, επιλέχθηκε η PHP για την υλοποίηση του συγκεκριμένου τμήματος της εφαρμογής για τους εξής λόγους:

- Είναι εύκολη στην εκμάθηση και χρήση γλώσσα και μπορεί να χρησιμοποιηθεί τόσο για μικρά όσο και για μεγάλης κλίμακας projects. Διαθέτει έναν τεράστιο αριθμό από βιβλιοθήκες αλλά και από επεκτάσεις (σε μορφή modules), οι οποίες επιτρέπουν λειτουργίες επεξεργασίας γραφικών, αποστολής email, δημιουργίας και επεξεργασίας αρχείων pdf, xml κτλ.
- Επειδή αποτελεί δυναμική γλώσσα προγραμματισμού η οποία μεταγλωττίζεται καθώς εκτελείται, υστερεί σε ταχύτητα σε σχέση με άλλες τεχνολογίες (βλ. JSP) παρόλα αυτά είναι κατά πολύ ταχύτερη σε σχέση με κλασσικές τεχνολογίες scripting στον server (π.χ. Common Gateway Interface scripting)
- Παρέχει μεγάλο αριθμό bindings για επικοινωνία με βάσεις δεδομένων και επιτυγχάνει άψογη συνεργασία με το σύστημα MySql.
- Επιτρέπει να την χρησιμοποίηση τεχνικών οντοκεντρικού προγραμματισμού, κάτι πάρα πολύ σημαντικό για την οργανωμένη συγγραφή και αξιοποίηση του κώδικα της εφαρμογής.
- Είναι πάρα πολύ διαδεδομένη γλώσσα στον τομέα της, χρησιμοποιείται στα περισσότερα internet sites. Αυτό έχει σαν αποτέλεσμα την ύπαρξη μιας μεγάλης και οργανωμένης κοινότητας χρηστών που παρέχει συμβουλές και βοήθεια καθώς και πάρα πολλά εργαλεία ανάπτυξης και αποσφαλμάτωσης του κώδικα.
- Αποτελεί open-source project το οποίο υποστηρίζεται από ένα μεγάλο αριθμό web servers και λειτουργικών συστημάτων.

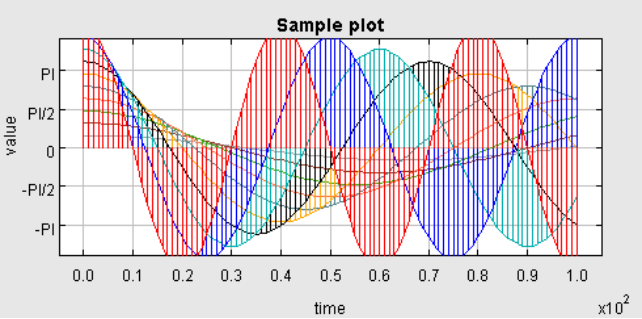
2.8 Τεχνικές απεικόνισης και διαχείρισης δισδιάστατων γραφικών στα πλαίσια ιστοσελίδων

Όπως έχει ήδη αναφερθεί, ένα πολύ σημαντικό μέρος της εφαρμογής αποτελεί η δημιουργία του επεξεργαστή (editor) δισδιάστατων σχεδίων κατόψεων. Αυτό

σημαίνει ότι θα πρέπει να υλοποιηθεί ένα σύστημα παρουσίασης και επεξεργασίας δισδιάστατων γραφικών μέσα στα πλαίσια της ιστοσελίδας. Επίσης θα πρέπει να υποστηρίζεται διαδραστικότητα με τα γραφικά αυτά. Παρακάτω θα παρουσιαστούν διάφορες τεχνολογίες, οι οποίες επιτρέπουν την υλοποίηση αυτού το τμήματος της εφαρμογής.

2.8.1 Java Applet με χρήση βιβλιοθήκης JAVA 2D

Η βιβλιοθήκη δισδιάστατων γραφικών της java προσφέρει αρκετές συναρτήσεις για την δημιουργία γραφικών στην οθόνη, από επίπεδο pixel μέχρι functions για την δημιουργία primitive σχημάτων, γεωμετρικούς μετασχηματισμούς κλπ. Η βιβλιοθήκη είναι προσβάσιμη από τα applets, τα οποία όπως είπαμε απαιτούν την ύπαρξη Java Runtime Environment στο σύστημα του χρήστη που επισκέπτεται τον ιστοτόπο της εφαρμογής (Εικόνα 17).



Εικόνα 17 - Γραφική απεικόνιση σημάτων χρησιμοποιώντας java 2d.

2.8.2 Flash

Η πλατφόρμα flash (Εικόνα 18) ενδείκνυται για την παρουσίαση διαδραστικών δισδιάστατων γραφικών καθώς αυτός ήταν ο σκοπός για τον οποίο δημιουργήθηκε αρχικά. Υποστηρίζει διαδικασίες για την δημιουργία και την επεξεργασία δυσδιάστατων διανυσματικών γραφικών καθώς επίσης και ένα σύστημα events και action listeners για την υποστήριξη διαδραστικότητας. Το Flash πετυχαίνει μεγάλη ταχύτητα στην γραφική αναπαράσταση αλλά απαιτεί την ύπαρξη plug-in. Επίσης αποτελεί κλειστή τεχνολογία. Περιλαμβάνει bindings για PHP ώστε να επιτυγχάνεται επικοινωνία με τον διακομιστή και ακολούθως με το σύστημα της βάσης δεδομένων. Η ταχύτητα εκτέλεσης που επιτυγχάνεται μέσω του plug-in του flash είναι υψηλή σε σχέση με άλλες υλοποιήσεις.

Εικόνα 18 - Χρησιμοποίηση flash για την δημιουργία δισδιάστατων διανυσματικών γραφικών στους χάρτες του yahoo maps.

2.8.3 Microsoft Silverlight

Η ανάλογη πρόταση του Flash από την Microsoft επιτρέπει την hardware-accelerated απεικόνιση δισδιάστατων γραφικών μέσα στα πλαίσια μιας σελίδας html, την υλοποίηση διαδραστικότητας μέσω scripting. Απαιτεί την ύπαρξη plugin.

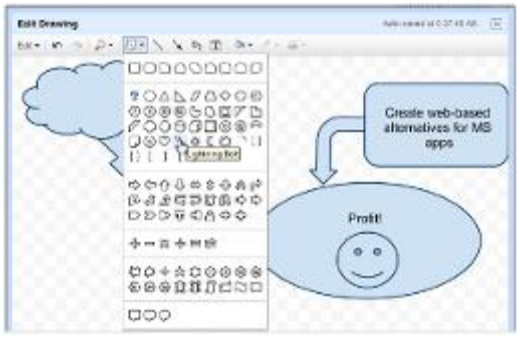
2.8.4 HTML CANVAS element

Στις τελευταίες εκδόσεις γνωστών φυλλομετρητών υποστηρίζεται ένα αντικείμενο που περιγράφεται στο specification της επερχόμενης έκδοσης HTML 5 – το canvas. Στην ουσία αποτελεί ένα component που εμφανίζει μια ορθογώνια περιοχή σχεδίασης στην ιστοσελίδα και παρέχει μέσω εντολών Javascript τον σχεδιασμό γραφικών raster όμως μορφής και όχι διανυσματικής. Δυστυχώς δεν βολεύει για την υλοποίηση της εφαρμογής μας καθώς τα σχέδια κατόψεων που θα δημιουργεί ο χρήστης απαιτούν διανυσματικά γραφικά στα οποία θα πρέπει να υπάρχει ευελιξία επιλογής και μετασχηματισμού σχημάτων. Αντίθετα το αντικείμενο canvas δουλεύει σε επίπεδο pixels και raster γραφικών και διατίθεται περισσότερο για την επεξεργασία π.χ. εικόνων και φωτογραφιών (Εικόνα 19).

Εικόνα 19 - – η εικόνα αυτή αποτελεί κομμάτι ιστοσελίδας στην οποία υλοποιείται η εφαρμογή paint χρησιμοποιώντας στο στοιχείο canvas.

2.8.5 SVG

Η τεχνολογία Scalable Vector Graphics αποτελεί μια ειδική διάλεκτο της γλώσσας XML (όπως π.χ. οι mathml, xhtml κτλ) για την περιγραφή διανυσματικών γραφικών. Τα αρχεία SVG επιτρέπουν την περιγραφή βασικών σχημάτων μέσω ειδικών tags (ορθογώνια, κύκλοι, πολύγωνα κτλ). Κάθε tag δέχεται έναν μεγάλο αριθμό από attributes τα οποία περιγράφουν παραμέτρους του κάθε σχήματος όπως η θέση του στον χώρο, η γωνία περιστροφής του, το χρώματα γεμίσματός του, το χρώμα του περιγράμματος, το πάχος του περιγράμματος κλπ. Επίσης, συστοιχίες σχημάτων μπορούν να ομαδοποιηθούν σε group, χρησιμοποιώντας ειδική tag και να εφαρμοστούν πάνω τους μετασχηματισμοί, αλλαγές χρωμάτων, κινήσεις. Επιπλέον, όντας ομαδοποιημένα μπορούν να χρησιμοποιηθούν σαν μοτίβα για γέμισμα άλλων σχημάτων κτλ. Η SVG χρησιμοποιεί πίνακες για γεωμετρικούς μετασχηματισμούς και επίσης περιλαμβάνει ειδικά tags για εφαρμογή filters πάνω στα γραφικά. Η SVG μπορεί να χρησιμοποιηθεί σαν αυτόνομο αρχείο (*.SVG) στο οποίο περιγράφονται τα γραφικά μέσω κειμένου (χρησιμοποιώντας tags και attributes). Ο φυλλομετρητής που διαβάζει το αρχείο αυτό θα πρέπει να υποστηρίζει το SVG specification για να παράγει τα δυσδιάστατα γραφικά στην οθόνη. Επίσης η SVG μπορεί να συμπεριληφθεί μέσα στο σώμα μιας σελίδας html χρησιμοποιώντας το ειδικό tag <SVG> (η μορφή αυτή ονομάζεται inline SVG). Επιπροσθέτως μέσα στα αρχεία της SVG μπορεί να συμπεριληφθεί κώδικας Javascript για τον προγραμματισμό της διαδραστικότητας. Αν χρησιμοποιείται inline SVG η διαδραστικότητα μπορεί να υλοποιηθεί από κώδικα Javascript που μπορεί να περιλαμβάνεται στο σώμα της ιστοσελίδας ή σε αναφερόμενο εξωτερικό αρχείο script. Στην τωρινή της υποστήριξη η SVG, ιδιαίτερα από τον Firefox παρέχει αξιοπρεπή ταχύτητα στην εμφάνιση των δισδιάστατων γραφικών η οποία όμως δεν φτάνει την ταχύτητα του flash. Στις πειραματικές δοκιμαστικές εκδόσεις του Firefox όμως όπου υποστηρίζεται Direct-X στο rendering της σελίδας η ταχύτητα απεικόνισης των γραφικών SVG ανεβαίνει κατακόρυφα (Εικόνα 20).



Εικόνα 20 – Χρησιμοποίηση SVG τεχνολογίας στο online εργαλείο σχεδίασης σχημάτων google draw το οποίο αποτελεί μέλος των Google docs.

2.8.6 Vector Markup Language

Βασίζεται στην XML και αναπτύχθηκε από την Microsoft σαν κάτι ανάλογο της SVG. Αντίθετα όμως με την SVG, που έχει προταθεί σαν στάνταρ από το W3C, η VML δεν έτυχε ανάλογης αποδοχής και η ανάπτυξή της σταμάτησε από το 1998. Υποστηρίζεται μόνο από Internet Explorer.

2.8.7 Αιτιολόγηση της επιλογής της γλώσσας SVG

Στα πλαίσια υλοποίησης του συγκεκριμένου τμήματος δισδιάστατων γραφικών της εφαρμογής, επιλέχθηκε η τεχνολογία SVG για τους εξής λόγους:

- Πρώτον γιατί αποτελεί native τεχνολογία που υποστηρίζεται εγγενώς από τους περισσότερους φυλλομετρητές, χωρίς να απαιτείται η ύπαρξη ενός plug-in όπως στις περιπτώσεις Flash, Silverlight και java applets
- Μπορεί να ενσωματωθεί μέσα στο σώμα της XHTML σελίδας και συγγενεύουν σε μορφή καθώς αποτελούν και οι 2 παράγωγα της XML. Η μία είναι περιγραφική γλώσσα αντικειμένων μέσα σε μια ιστοσελίδα και η άλλη περιγραφική γλώσσα διανυσματικών γραφικών. Το scripting γίνεται με την γλώσσα Javascript την οποία ήδη χρησιμοποιούμε στο κύριο μέρος εκτέλεσης της εφαρμογής μας.
- Χρησιμοποιεί ξεχωριστές οντότητες, για κάθε σχήμα και γραφικό που παρουσιάζεται στην οθόνη και επιτρέπεται η επί μέρους επεξεργασία του και η εφαρμογή μετασχηματισμών, χωρίς να χρειάζεται ανανέωση όλου του πλαισίου γραφικών όπως π.χ. γίνεται στο canvas element το οποίο ανανεώνει όλο το bitmap των γραφικών.
- Διαθέτει ευέλικτο μοντέλο διαχείρισης events μέσω ποντικιού και μέσω πληκτρολογίου.
- Επειδή αποτελεί μορφή xml, μπορεί εύκολα να χρησιμοποιηθεί για μεταφορά και αποθήκευση δεδομένων προς το διακομιστή.

2.9 Τεχνολογίες απεικόνισης τρισδιάστατων γραφικών μέσα σε περιβάλλον φυλλομετρητή

Το δεύτερο, πιο σημαντικό τμήμα της εφαρμογής, είναι η παρουσίαση της τρισδιάστατης σκηνής και η περιήγηση του χρήστη σε αυτή. Παρακάτω παρουσιάζονται τεχνολογίες που επιτρέπουν την απεικόνιση τρισδιάστατων γραφικών σε περιβάλλον φυλλομετρητή.

2.9.1 Java Applet με χρήση βιβλιοθήκης Java 3D

Όπως και με τα δισδιάστατα γραφικά θα μπορούσε να χρησιμοποιηθεί η βιβλιοθήκη java3d μέσα στα πλαίσια ενός applet. Η συγκεκριμένη βιβλιοθήκη επιτρέπει την δημιουργία πολύπλοκων τρισδιάστατων σκηνών οι οποίες είναι επιταχυμένες από την κάρτα γραφικών (hardware accelerated). Παρέχει δενδρικό σύστημα (scene-graph) για διαχείριση της γεωμετρίας και εκτελείται βασιζόμενη στις τεχνολογίες DirectX ή Opengl. Δυστυχώς όμως συνοδεύεται από τα μειονεκτήματα που αναφέρθηκαν ως τώρα για την χρήση applets στην υλοποίηση της εφαρμογής (Εικόνα 21).

Εικόνα 21 – Runescape ένας online τρισδιάστατος κόσμος παιχνιδιού υλοποιημένος αποκλειστικά με java 3d.

2.9.2 Flash player 10

Στις τελευταίες εκδόσεις του Flash, η δημιουργός εταιρία Adobe, πρόσθεσε δυνατότητες 3d μετασχηματισμών σε αντικείμενα και εφαρμογή textures πάνω σε αυτά. Δυστυχώς οι λειτουργίες αυτές, βρίσκονται ακόμα σε πρώιμο στάδιο ώστε να υποστηρίξουν μια περίπλοκη 3d εφαρμογή. Για αυτό έχουν δημιουργηθεί τρισδιάστατες μηχανές γραφικών οι οποίες εκτελούνται χρησιμοποιώντας την τεχνολογία flash. Μερικές από αυτές τις μηχανές παρουσιάζονται παρακάτω.

2.9.3 PaperVision

Αποτελεί ένα σύνολο κλάσεων και συναρτήσεων σε Action Script οι οποίες δίνουν την δυνατότητα για δημιουργία τρισδιάστατων αντικειμένων. Στην ουσία η μηχανή εξομοιώνει την ύπαρξη τρισδιάστατων σκηνών και κατόπιν με κατάλληλους μετασχηματισμούς και optimized τεχνικές μεταφράζει την απεικόνιση χρησιμοποιώντας όμως γραφικά τα οποία δεν είναι επιταχυμένα από την άρτα γραφικών. Αυτό έχει σαν αποτέλεσμα μικρή ταχύτητα στην αναπαράσταση και μεγάλη δέσμευση υπολογιστικών πόρων (Εικόνα 22).

Εικόνα 22 – Δείγμα τρισδιάστατων γραφικών από την μηχανή PaperVision (βασισμένη σε τεχνολογία flash).

2.9.4 Away3d

Παρόμοια μηχανή γραφικών με την PaperVision3d. Στηρίζεται πάνω στην τεχνολογία Flash και χρησιμοποιεί διαδικασίες που στην ουσία αναπαριστούν τα γραφικά χωρίς να χρησιμοποιούν τρισδιάστατη επιτάχυνση. (Εικόνα 23).

Εικόνα 23 – Δείγμα τρισδιάστατων γραφικών από την μηχανή Away3d (βασισμένη σε τεχνολογία flash).

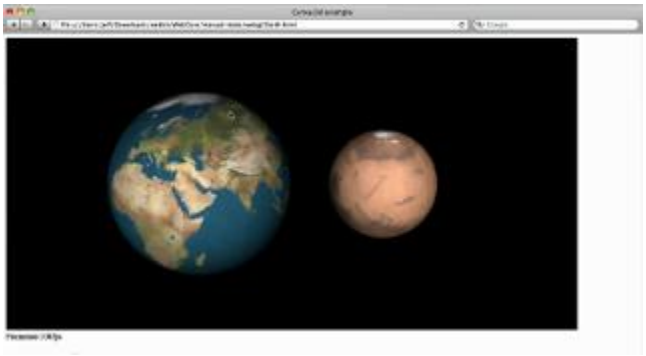
2.9.5 Unity 3d Player

Η unity3d αποτελεί μια ολοκληρωμένη 3d μηχανή γραφικών η οποία χρησιμοποιείται για την παραγωγή παιχνιδιών. Διαθέτει περιβάλλον ανάπτυξης με editor, scripting files και παραμέτρους. Η μεταγλώττιση γίνεται για την δημιουργία desktop εφαρμογών οι οποίες όμως μπορούν εύκολα να μεταγλωττιστούν για χρήση σε σελίδες του παγκόσμιου ιστού. Η απεικόνιση των γραφικών είναι επιταχυμένη (hardware accelerated) οπότε επιτυγχάνεται υψηλή ταχύτητα αποδεσμεύοντας τον κεντρικό επεξεργαστή. Η unity χρησιμοποιεί ένα ειδικό plug-in για την εκτέλεση των εφαρμογών της σε περιβάλλον φυλλομετρητή, to unity 3d player (Εικόνα 24).

Εικόνα 24 - Εκτέλεση τρισδιάστατου παιχνιδιού μέσα από ιστοσελίδα χρησιμοποιώντας το unity 3d player.

2.9.6 Webgl

Η Webgl αποτελεί ένα πλαίσιο του αντικειμένου <canvas> της html (εισαχθέν στην HTML 5), η οποία προσφέρει διεπαφή προγραμματισμού 3d εφαρμογών μέσα σε περιβάλλον φυλλομετρητή χωρίς την χρήση plug-ins. Η Webgl βασίζεται στην Opengl ES 2.0, η οποία αποτελεί ένα υποσύνολο της γενικής μορφής Opengl που συναντάμε στην ανάπτυξη των desktop εφαρμογών. Η Webgl περιλαμβάνει ρουτίνες σε Javascript για τον προγραμματισμό των 3d γραφικών και χρησιμοποιεί το Document Object Model σε συνδυασμό με το αντικείμενο canvas, για την απεικόνισή τους μέσα σε πλαίσια ιστοσελίδων (Εικόνα 25). Υποστηρίζεται από τους πιο διαδεδομένους φυλλομετρητές σήμερα όπως Mozilla Firefox, Opera Browser, Apple Safari και Google Chrome.



Εικόνα 25 – 3d γραφικά μέσα σε φυλλομετρητή με την χρήση της τεχνολογίας Webgl.

2.9.7 Google O3d

Το O3D αποτελεί την προσπάθεια της Google, για υποστήριξη απεικόνισης τρισδιάστατων γραφικών στο internet. Προς το παρόν, δεν αποτελεί native κομμάτι των φυλλομετρητών και απαιτεί την εγκατάσταση ειδικού plug-in. Χρησιμοποιεί ειδικές Javascript βιβλιοθήκες για τον προγραμματιστικό κομμάτι. Περιλαμβάνει δικό

του scene-graph. Το scene-graph είναι μια δενδρική δομή δεδομένων που αποθηκεύει λεπτομερείς πληροφορίες για την ιεραρχία της τρισδιάστατης γεωμετρίας μιας σκηνής. Υποστηρίζει τελευταίες τεχνολογίες στην απεικόνιση τρισδιάστατων γραφικών όπως προγραμματισμός με shaders, post processing effects κτλ. Περιλαμβάνει βοηθητικές βιβλιοθήκες για την δήλωση και δημιουργία απλών γεωμετρικών σχημάτων (π.χ. κύβοι, σφαίρες, επίπεδα κτλ), ειδικές συναρτήσεις για το αυτόματο φόρτωμα τρισδιάστατων μοντέλων από αρχεία και συναρτήσεις για τον προσδιορισμό materials και textures για τα αντικείμενα (Εικόνα 26). Περιλαμβάνει συναρτήσεις για την διαχείριση των μετασχηματισμών της 3d γεωμετρίας, για τον ορισμό καμερών κτλ. Τα αρχεία που χρησιμοποιεί το Google o3d περιορίζονται στο πρότυπο collada και Google sketch up. Εν τέλει, ανακοινώθηκε ότι το Google O3D, δεν θα αποτελεί αυτόνομη τεχνολογία (όπως προοριζόταν αρχικώς) αλλά θα εκτελείται χρησιμοποιώντας την Webgl.

Εικόνα 26 – Εμφάνιση τρισδιάστατων γραφικών πραγματικού χρόνου μέσα σε browser με χρήση google O3D.

2.9.8 X3D

Αποτελεί μια γλώσσα περιγραφής τρισδιάστατων γεωμετρικών σκηνών η οποία βασίζεται στο πρότυπο της XML. Η X3D είναι ο συνεχιστής της VRML η οποία χρησιμοποιούνταν τα πρώτα χρόνια του ιντερνέτ για την περιγραφή τρισδιάστατων εικονικών κόσμων μέσα στους φυλλομετρητές (Εικόνα 27). Στην X3D χχρησιμοποιούνται απλά tags για να περιγράψουν βασικά τρισδιάστατα σχήματα όπως κύβοι, κύλινδροι, επίπεδα κτλ. Επίσης υπάρχουν συγκεκριμένα tags για την περιγραφή πιο περίπλοκης γεωμετρίας όπως 3d meshes, nurb surfaces κλπ. Η X3D αποτελεί standard και γίνεται μια προσπάθεια ενσωμάτωσης της εγγενώς μέσα στις σελίδες HTML 5. Προς το παρόν, η ενσωμάτωση X3D σκηνών σε περιβάλλων φυλλομετρητή γίνεται με την ύπαρξη plug-ins και συγκεκριμένα VRML/X3D players όπως ο Octaga Player, ο BS Contact Player κτλ. Το interactivity των σκηνών επιτυγχάνεται με scripting μέσω της γλώσσας Javascript.

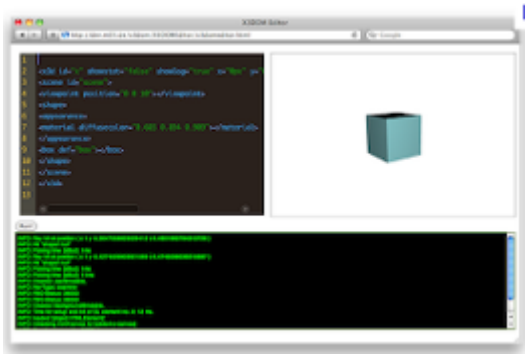
Εικόνα 27 – Αρχιτεκτονική απεικόνιση σταδίου με χρήση X3D.

2.9.9 Αιτιολόγηση επιλογής της τεχνολογίας X3D

Στην συγκεκριμένη εφαρμογή της διπλωματικής, από όλες τις τεχνολογίες τρισδιάστατης αναπαράστασης επιλέχθηκε η **X3D**, γιατί αποτελεί ένα ισχυρό στάνταρ το οποίο διαμορφώνεται εδώ και χρόνια (από την αρχική ύπαρξη της VRML στα μέσα της δεκαετίας του 90). Η περιγραφή της τρισδιάστατης σκηνής γίνεται σε γλώσσα που αποτελεί επέκταση της XML, άρα μπορούν να χρησιμοποιηθούν τεχνικές προγραμματισμού DOM καθώς και βιβλιοθήκες που ήδη χρησιμοποιούνται και σε άλλα τμήματα της εφαρμογής. Η διαδραστικότητα των τρισδιάστατων αντικειμένων υλοποιείται με Javascript μια γλώσσα προγραμματισμού που ήδη χρησιμοποιείται κατά κύριο λόγω στα πλαίσια της εφαρμογής.

2.9.10 X3Dom

Αποτελεί προσπάθεια του Fraunhofer Institute, για την υποστήριξη απεικόνισης τρισδιάστατων σκηνών που περιγράφονται σε μορφή X3D, μέσα στο περιβάλλον του φυλλομετρητή, αλλά χωρίς την χρήση plug-ins. Εκτελείται χρησιμοποιώντας την τεχνολογία Webgl. Αποτελείται από μια βιβλιοθήκη Javascript, η οποία περιλαμβάνει δομές που διαβάζουν την δομή του X3D αρχείου και μέσα από κατάλληλες διαδικασίες καλούνται εντολές Webgl, οι οποίες δημιουργούν αντίστοιχες τρισδιάστατες γεωμετρίες. Η τεχνολογία αυτή είναι ακόμα σε πειραματικό στάδιο και υποστηρίζεται από ορισμένες δοκιμαστικές εκδόσεις κάποιων γνωστών browsers (Εικόνα 28).



Εικόνα 28 – Χρήση του X3DDom σε περιβάλλον φυλλομετρητή.

ΚΕΦΑΛΑΙΟ 3
ΤΕΧΝΟΛΟΓΙΚΗ ΒΑΣΗ

3.1 Εισαγωγή

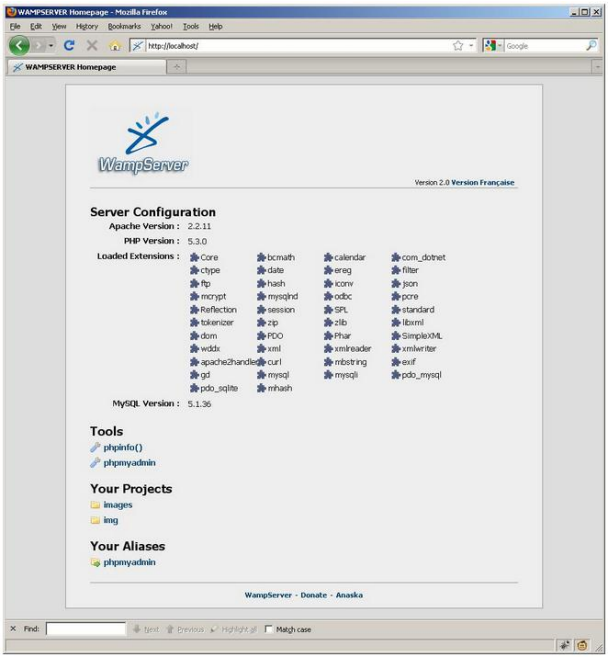
Στο προηγούμενο κεφάλαιο έγινε αναφορά στην σχετική έρευνα που πραγματοποιήθηκε για τις τεχνολογίες εκείνες, οι οποίες θα βοηθούσαν στην υλοποίηση συγκεκριμένων τμημάτων της εφαρμογής. Αφού παρουσιάστηκε ένας αριθμός διαφορετικών τεχνολογιών και μεθοδολογιών ανάπτυξης, εν τέλει, υιοθετήθηκαν και χρησιμοποιήθηκαν συγκεκριμένες από αυτές. Στο παρόν κεφάλαιο θα γίνει αναλυτική περιγραφή την αρχιτεκτονικής και της λειτουργίας όλων των τεχνολογιών, των εργαλείων και των μεθοδολογιών που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής.

3.2 Περιβάλλον Διακομιστή Ιστού (Web Server)

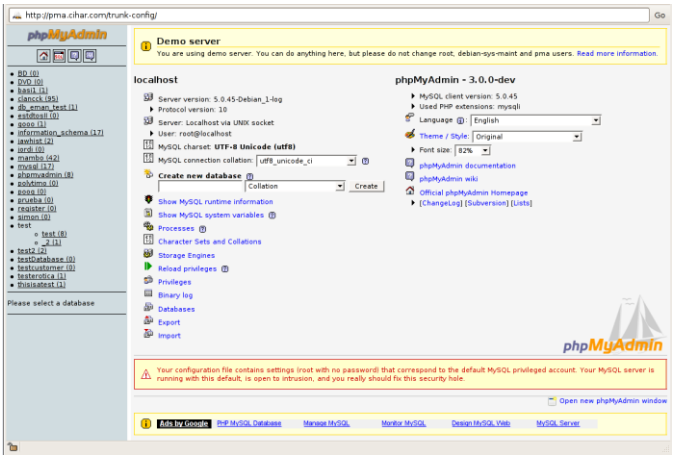
Η πιο θεμελιώδη πτυχή της υλοποίησης είναι ο καθορισμός του περιβάλλοντος μέσα στο οποίο θα αναπτυχθεί, θα εδρεύει και θα εκτελείται η εφαρμογή. Όπως προαναφέρθηκε η εφαρμογή δεν έχει desktop μορφή, δηλαδή δεν αποτελεί μεταγλωττισμένο προϊόν του κώδικα μιας συγκεκριμένης γλώσσας με έτοιμο εκτελέσιμο αρχείο, το οποίο εγκαθίσταται στο λειτουργικό του χρήστη και εκτελείται από εκεί. Η εφαρμογή, έχοντας διαδικτυακή μορφή και μάλιστα επειδή το περιεχόμενό της παρουσιάζεται με την μορφή δυναμικών ιστοσελίδων πρέπει να εδρεύει σε έναν Εξυπηρετητή ιστού. Εκεί θα είναι τοποθετημένα όλα τα αρχεία της. Εκεί θα βρίσκεται και η βάση δεδομένων που θα αποθηκεύει όλες τις απαραίτητες πληροφορίες που θα πρέπει να κρατηθούν. Ο διακομιστής θα εκτελεί το κομμάτι εκείνο που πρέπει να εκτελεστεί στην πλευρά του (π.χ. βάση δεδομένων, PHP scripts) και θα μεταφέρει στον φυλλομετρητή του πελάτη χρήστη τις σελίδες html και τον κώδικα Javascript που πρέπει να εκτελεστεί σε περιβάλλον client. Άρα συνοψίζοντας μπορούμε να πούμε ότι η εφαρμογή μας εκτελείται ταυτόχρονα σε δύο σημεία. Ένα κεντρικό κομμάτι της στο εξυπηρετητή ιστού και ένα άλλο κομμάτι της σε κάθε φυλλομετρητή χρήστη που είναι συνδεδεμένος στον ιστοτόπο της και την χρησιμοποιεί. Για διακομιστή ιστού όπως προαναφέρθηκε, έγινε η επιλογή του Apache Http Server. Για γλώσσα εκτέλεσης κώδικα στο περιβάλλον εξυπηρετητή, επιλέχθηκε η PHP, η οποία υποστηρίζεται κατευθείαν από το περιβάλλον του Apache. Για διακομιστής βάσεως δεδομένων επιλέχθηκε ο MySQL Server, ο οποίος συνεργάζεται άψογα με τον Apache, με γέφυρα επικοινωνίας την PHP. Αυτό το τρίπτυχο (Apache – MySQL – PHP) είναι πολύ διαδεδομένο στην ανάπτυξη ιστοσελίδων και διαδικτυακών εφαρμογών. Ειδικότερα, χρησιμοποιούνται τα δύο

ακρωνύμια W.A.M.P και L.A.M.P, τα οποία αναφέρονται αντίστοιχα στον συνδυασμό Windows – Apache – Mysql – PHP και στον συνδυασμό Linux – Apache – MySql – PHP. Στην πρώτη περίπτωση υπάρχει πακέτο αυτόματης εγκατάστασης και του Server Apache, με το περιβάλλον και τα modules της PHP, αλλά και του MySql Server σε λειτουργικό Windows, ενώ στην δεύτερη η εγκατάσταση γίνεται σε λειτουργικό Linux. Για την υλοποίηση της συγκεκριμένης εφαρμογής επιλέχθηκε ο Wamp Server 2.0 για περιβάλλον Windows 7. Το πακέτο περιέχει οδηγό (wizard) αυτόματης εγκατάστασης όλων των περιεχομένων και αυτόματης ρύθμισής τους (Εικόνα 29).

Εικόνα 29 – Ο οδηγός εγκατάστασης του Wamp server.



Εικόνα 30 - Η σελίδα σωστής λειτουργίας του Wamp server.



Εικόνα 31 – Το εργαλείο PhpMyAdmin.

Παράλληλα γίνεται εγκατάσταση του εργαλείου PhpMyAdmin (Εικόνα 30), για γραφική επικοινωνία με τον MySql Server. Στον σκληρό δίσκο δημιουργείται ένας φάκελος Wamp ο οποίος περιέχει ένα tray tool το οποίο εμφανίζεται στην μπάρα έναρξης των windows και παρέχει συντομεύσεις στις πιο κύριες λειτουργίες του server. Μέσα στον φάκελο Wamp περιέχεται ο φάκελος bin με περιεχόμενα τους φακέλους Apache, MySql και PHP στους οποίους είναι αποθηκευμένα τα απαραίτητα αρχεία κάθε τεχνολογίας. Σε κάθε Webserver υπάρχει ένας φάκελος , ο επονομαζόμενος webroot στον οποίο τοποθετούνται οι ιστοσελίδες που θα γίνουν διαθέσιμες στον παγκόσμιο ιστό. Στον webroot κατάλογο μπορούν να δημιουργηθούν υποφάκελοι ένας για κάθε διαφορετικό site που φιλοξενείται στον διακομιστή. Στο πακέτο Wamp server ο webroot φάκελος είναι στο εξής path: "…/wamp/www". Μέσα λοιπόν στον φάκελο «www» δημιουργήθηκε έναν φάκελος Homedraw για να φιλοξενήσει τα αρχεία του ιστοτόπου της εφαρμογής μας. Ο έλεγχος της σωστής λειτουργίας του ιστοτόπου της εφαρμογής θα γίνει τοπικά, στον υπολογιστή όπου γίνεται η ανάπτυξη και στον οποίο έχει εγκατασταθεί το πακέτο wamp. Η τοπική πρόσβαση στον Apache Server γίνεται ανοίγοντας τον φυλλομετρητή και πληκτρολογώντας την διεύθυνση http://localhost (Εικόνα 30). Για να έχουμε πρόσβαση στον ιστοτόπο της εφαρμογής μας πληκτρολογούμε http://localhost/Homedraw/. Θα πρέπει να δημιουργηθεί μια αρχική ιστοσελίδα για την εφαρμογή μας, η οποία συνήθως ορίζεται ως index.htm. Άρα το τελικό url για την έναρξη της web εφαρμογής τοπικά θα είναι http://localhost/Homedraw/index.htm.

3.2.1. Ρυθμίσεις PHP

Το tray tool του Wamp server παρέχει συντόμευση στις ρυθμίσεις περιβάλλοντος της PHP. Στην ουσία επιτρέπεται η αλλαγή των παραμέτρων της γλώσσας στο αρχείο PHP.ini. Μια πρώτη ένδειξη για το ότι όλα λειτουργούν σωστά είναι να δημιουργήσουμε ένα απλό PHP script αρχείο το οποίο περιλαμβάνει την εντολή phpInfo(); (Εικόνα 32).

Εικόνα 32 – Η παραγόμενη σελίδα από την κλήση της συνάρτησης phpInfo();.

Η phpInfo() είναι μια συνάρτηση, η οποία αυτόματα δημιουργεί μια σελίδα με ένα μεγάλο αριθμό πληροφοριών, για την τρέχουσα κατάσταση του περιβάλλοντος PHP στον εξυπηρετητή ιστού. Εκεί περιλαμβάνονται πληροφορίες για τις επιλογές μεταγλώττισης της PHP και των επεκτάσεών της, πληροφορίες για την έκδοση PHP κτλ.

3.2.2. Ρυθμίσεις MySql Server

Από το tray tool του Wamp Server εκτελούμε το command line administrative tool του MySql server και δίνουμε root για username και password (Έστω ότι ο κωδικός μας είναι ο 12345) για να μπούμε στο σύστημα βάσεων δεδομένων. Εκεί εκτελούμε την εντολή **show databases;** για να δούμε λίστα με τις υπάρχουσες βάσεις δεδομένων που υπάρχουν κατά την εγκατάσταση και για να επαληθεύσουμε ότι ο server είναι τρέχει ορθά. Για να δούμε ότι υπάρχει αρμονική συνεργασία μεταξύ των τριών τεχνολογιών apache – Mysql – PHP δημιουργούμε ένα μικρό script file στον κατάλογο "www" του apache με την ονομασία mysql.PHP το οποίο περιέχει τον παρακάτω κώδικα:

```
<?
// connect to the server:
$cn = mysql_connect("127.0.0.1","root",”12345");
// run a simple query
$sql = "SELECT 'done' as my_field LIMIT 1";
$result = mysql_query($sql,$cn);

if($result)
{
// if it worked, print the result to screen
echo mysql_result($result,"my_field");
} else {
// otherwise, either the server isn't running
// or the username/password are wrong
echo mysql_error()."You should see an error message above you?";
```

```
}
?>
```

Εκτελούμε το παραπάνω script δίνοντας στον browser το url : http://localhost/mysql.PHP

Αν στην σελίδα που θα φορτωθεί εμφανιστεί το μήνυμα done! Τότε σημαίνει ότι η επικοινωνία με τον Apache Http Server και με τον MySQL server μέσω PHP πραγματοποιήθηκε με επιτυχία. Ακολουθεί η περιγραφή των υπόλοιπων τεχνολογιών της εφαρμογής.

3.3 HTML και XHTML

Η HTML αποτελεί μια γλώσσα σήμανσης (markup language), δηλαδή μια γλώσσα η οποία χρησιμοποιείται για να περιγράψει και να διαχωρίσει, τα διαφορετικά μέρη, ενός εγγράφου ιστοσελίδας. Παρέχει μέσα για την δημιουργία ορθά δομημένων εγγράφων υποδηλώνοντας σημασιολογικά στοιχεία κειμένου όπως παράγραφοι, επικεφαλίδες, λίστες, σύνδεσμοι, εισαγωγικά κτλ. Επίσης επιτρέπει την ενσωμάτωση εικόνων και αντικειμένων όπως πλαίσια κειμένου, combo boxes, κουμπιά κτλ τα οποία μπορούν να χρησιμοποιηθούν για την δημιουργία interactive φορμών. Μια ιστοσελίδα σε γλώσσα HTML αποτελείται από στοιχεία κειμένου (π.χ. επικεφαλίδες, υποσημειώσεις, παράγραφοι, επισημάνσεις, παραπομπές, σύνδεσμοι, λίστες κτλ), από στοιχεία πολυμέσων (εικόνες, γραφήματα, βίντεο κτλ) , στοιχεία φορμών (κουμπιά, πλαίσια κειμένου, combo boxes κτλ) τα οποία ονομάζονται html elements. Η δήλωση ενός html element αποτελείται από τα εξής βασικά συστατικά:

Α) Ένα ζεύγος **tags** του στοιχείου: η tag ανοίγματος και η tag κλεισίματος.

Β) Ένα σύνολο από **attributes** η οποίες δηλώνονται μέσα στα tags του στοιχείου
Και τέλος το περιεχόμενο του στοιχείου προς παρουσίαση στην οθόνη του φυλλομετρητή το οποίο μπορεί να έχει είτε μορφή κειμένου, είτε μορφή γραφικών.

Ένα **html** στοιχείο είναι οτιδήποτε περιλαμβάνεται μεταξύ των δύο tags Η **tag** του στοιχείου στην ουσία είναι το όνομα του στοιχείου εσώκλειστο μεταξύ δύο αγκυλών. Η **tag** κλεισίματος περιλαμβάνει μπροστά από το όνομα του στοιχείου μια κάθετο. Παραδείγματος χάριν έστω ότι θέλουμε να περιγράψουμε την εξής παράγραφο σε μορφή html μέσα σε μια ιστοσελίδα :

```
The quick brown fox jumps over the lazy dog
```

Πρώτα θα πρέπει να επιλέξουμε το σωστό tag που περιγράφει μια παράγραφο και αυτό είναι το (και το για tag κλεισίματος). Ανάμεσά τους θα περιλαμβάνεται το περιεχόμενο της παραγράφου δηλ:

```
<p> The quick brown fox jumps over the lazy dog </p>
```

Μέσα στα tags ανοίγματος μπορούν να υπάρχουν και attributes που περιγράφουν παραμέτρους που επηρεάζουν το στοιχείο που περιγράφεται. Π.χ. στην περίπτωση περιγραφής μιας εικόνας μέσα σε μια ιστοσελίδα τα tags που θα χρησιμοποιηθούν είναι τα εξής `<img> </img>`. Παρατηρούμε ότι ανάμεσά τους δεν υπάρχει περιεχόμενο καθώς το περιεχόμενο για το στοιχείο της εικόνας ορίζεται από εξωτερικό αρχείο (π.χ. αρχείο εικόνας jpg, gif κτλ) Η αναφορά στο εξωτερικό αρχείο εικόνας θα γίνει ως attribute μέσα στο tag ανοίγματος. Πχ έστω ότι το path για το αρχείο εικόνας που θέλουμε να παρουσιάσουμε είναι το εξής "../img/summer/beach.jpg". Τότε η τελική μορφή επισήμανσης του στοιχείου στην html γίνεται :

```
<img src="…/img/summer/beach.jpg" > </img>
```

Η attribute src δέχεται για παράμετρο το path η το url αναφοράς στην εικόνα που θέλουμε να παρουσιάσουμε μέσα στην ιστοσελίδα μας. Αν θα θέλαμε να καθορίσουμε το πλάτος και το μήκος της εικόνας, θα μπορούσαμε να προσθέσουμε 2 ακόμα attributes στο tag ανοίγματος (τα width και height) ως εξής:

```
< img src="…/img/summer/beach.jpg" width="100px" height="200px">
</img>
```

Παρατηρούμε ότι τα attributes χωρίζονται μεταξύ τους με κενό. Επίσης σε κάθε attribute η δήλωση της τιμής γίνεται με "=" και μετά με την τιμή μέσα σε εισαγωγικά. Μπορούμε να συνοψίσουμε την γενική μορφή ενός html element ως εξής:

```
<tag attribute1="value1" attribute2="value2" attribute3="value3">
περιεχόμενο προς παρουσίαση </tag>
```

Επίσης ένα html element μπορεί να περιλαμβάνει μέσα στο περιεχόμενό του ένα άλλο html element. Π.χ. μια παράγραφος μέσα στο περιεχόμενο κείμενο μπορεί να περιλαμβάνει μια εικόνα δηλ:

```
<p> The Quick Brown Fox<img src="…/img/fox.jpg"></img> Jumps Over The
Lazy Dog</p>
```

Στην παραπάνω δήλωση παρατηρούμε ότι μετά την λέξη Fox στην παράγραφο, περιλαμβάνεται η δήλωση μιας εικόνας, η οποία θα εμφανιστεί σαν περιεχόμενο της παραγράφου ανάμεσα από το κείμενο. Ένα άλλο παράδειγμα έμφωλιασμένων html elements αποτελούν οι λίστες. Μια μορφή λίστας στην γλώσσα html είναι η unordered list (λίστα χωρίς ταξινόμηση). Η λίστα αυτή, περιλαμβάνει list items τα οποία σαν περιεχόμενο μπορούν να έχουν κείμενο η οποιοδήποτε html στοιχείο. Τα list items αποτελούν στοιχείο της html και δηλώνονται ως εξής:

```
<li> list item content </li>
```

Η unordered list δηλώνεται με το εξής tag:

```
<ul> unordered list content </ul>
```

Μια λίστα με 5 list items κειμένου θα μπορούσε να δηλωθεί ως εξής:

```
<ul>
<li> content of first item </li>
<li> content of second item </li>
<li> content of third item </li>
<li> content of fourth item </li>
<li> content of fifth item </li>
</ul>
```

Παρατηρούμε τα εμφωλευμένα list items μέσα στο περιεχόμενο του unordered list item. Κάθε έγγραφο ιστοσελίδας σε γλώσσα html απαιτεί την ύπαρξη κάποιων συγκεκριμένων tags που καθορίζουν την βασική δομή του. Καταρχήν υπάρχει το HTML element που εσωκλείει μέσα του όλο το περιεχόμενο της σελίδας, δηλ όλος ο κώδικας html περιέχεται μέσα στα tags <html></html>. Το html element χρησιμοποιείται για να αναγνωρίζει ο φυλλομετρητής ότι το περιεχόμενο που εσωκλείεται αποτελεί περιγραφή ιστοσελίδας σε γλώσσα html. Το στοιχείο <html> αποτελεί το root element του εγγράφου. Μέσα στο html element υπάρχουν εμφωλευμένα αρκετά στοιχεία όπως το head element. Το head element αποτελεί «επικεφαλίδα» του εγγράφου και περιλαμβάνει στοιχεία επεξεργασίας του html αρχείου όπως και metadata.Στο head element περιλαμβάνεται εμφωλευμένο title element, το οποίο περιέχει τον τίτλο τις ιστοσελίδας. Επίσης, στο head element περιλαμβάνονται αναφορές εξωτερικών αρχείων Javascript και cascading style sheets. Ένα άλλο βασικό στοιχείο το οποίο περιλαμβάνεται μέσα στο html element είναι το body element το οποίο αποτελεί «δοχείο» για όλα εκείνα τα στοιχεία του περιεχομένου της σελίδας που είναι εμφανίσιμα στην οθόνη του φυλλομετρητή. Το περιεχόμενο ενός βασικού αρχείο html με την ιεραρχία των βασικών elements περιγράφεται στον παρακάτω κώδικα.

```
<html>

<head>
<title> Title of Html Document </title>
</head>

<body>
Elements and Content to be Rendered by the browser
</body>

</html>
```

Επίσης για κάθε element της html γλώσσας υπάρχουν κάποια βασικά **attributes** τα οποία περιγράφονται παρακάτω:

- **Id**. Η Id attribute προσδίδει στο συγκεκριμένο στοιχείο ένα αναγνωριστικό όνομα, το οποίο είναι μοναδικό σε όλο το μήκος και πλάτος του εγγράφου html. Το αναγνωριστικό όνομα αυτό, μπορεί να χρησιμοποιηθεί από τα

Cascading Style Sheets για να γίνει αναφορά στο συγκεκριμένο html element και να αλλαχθεί η εμφάνισή του, είτε από script της Javascript για να υλοποιηθούν διεργασίες που αφορούν μόνο το συγκεκριμένο στοιχείο.

- **Class**. Η Class attribute χρησιμοποιείται για ομαδοποίηση html στοιχείων. Παραδείγματος χάριν μπορούμε να ορίσουμε σε ορισμένες παραγράφους της ιστοσελίδας μας την class attribute να ισούται με "attention_text". Αντίστοιχα, στα Cascading Style Sheets που περιγράφουν την εμφάνιση της ιστοσελίδας μας, μπορούμε να ορίσουμε έναν κανόνα, που να περιγράφει ότι όλα τα στοιχεία της κλάσης "attention_text" θα εμφανίζονται με έντονα και κόκκινου χρώματος γράμματα.

- **Title**. Η attribute title (να μην μπερδευτεί με το element title), μπορεί να τοποθετηθεί σε κάθε στοιχείο και να προσάψει μια γενική περιγραφή, η οποία σε ορισμένους browsers εμφανίζεται με την μορφή tooltip, όταν τοποθετούμε τον κέρσορα του ποντικιού πάνω από το συγκεκριμένο στοιχείο.

- **Lang**. Η attribute lang χρησιμοποιείται για να αναφέρει την γλώσσα του περιεχομένου ενός στοιχείου όταν εκείνη διαφέρει από την κύρια γλώσσα του εγγράφου.

Πέρα από τα βασικά Html Elements που περιγράψαμε παραπάνω, υπάρχουν διάφορα άλλα τα οποία τοποθετούνται στο body element μιας σελίδας. Κάποια από αυτά περιγράφονται παρακάτω:

- … Για την περιγραφή των παραγράφων κειμένου.

- <h1>…</h1>, <h2>…</h2>, <h3>…</h3>, <h4>…</h4> για την περιγραφή διάφορων επικεφαλίδων κειμένου.

- …, …, <dl>…</dl> για την δημιουργία λιστών.

- … ένα πολύ σημαντικό element για την διάταξη των ιστοσελίδων, το οποίο δεν προσδίδει σημασιολογικό διαχωρισμό, αλλά δομικό, καθώς χωρίζει την σελίδα σε αυτόνομα τμήματα εγγράφου, στα οποία μπορούν να αποδοθούν διαφορετικά στυλ εμφάνισης και διάταξης μέσω css και java. Συνήθως τα div elements χρησιμοποιούνται για την δημιουργία στηλών, footers, headers, layers κτλ μέσα σε μια σελίδα html.

Η Html επίσης επιτρέπει την ενσωμάτωση scripts σε γλώσσες όπως η Javascript, τα οποία επηρεάζουν την λειτουργία και την διαδραστικότητα μιας σελίδας. Ο κώδικας Javascript μπορεί να αποτελεί κομμάτι της σελίδας και να συμπεριλαμβάνεται ανάμεσα στα tags <script> … </script>, ή να βρίσκεται σε εξωτερικό αρχείο Javascript με κατάληξη *.js, το οποίο μπορεί να γίνει αναφορά μέσω της attribute src του στοιχείου script. Τέλος η Html μπορεί να περιλαμβάνει Cascade Styling Sheets τα οποία καθορίζουν την εμφάνιση και την διάταξη του κειμένου και των άλλων

στοιχείων μέσα στις ιστοσελίδες. Οι κανόνες css μπορεί είτε να αποτελούν κομμάτι της σελίδας και περικλείονται ανάμεσα στα tags <style> … </style> είτε να δηλώνονται σε εξωτερικό αρχείο με κατάληξη *.css. Η αναφορά στο εξωτερικό αρχείο μπορεί να γίνει με το element <link>….</link> και ρυθμίζοντας τις εξής attributes του:

- Rel=”stylesheet”
- Href=”…/css/stylesheet1.css/” – path προς το εξωτερικό αρχείο css.
- Type=”text/css”

Το World Wide Web Consortium, το οποίο είναι υπεύθυνο για την διατήρηση των στάνταρ και της Html και του CSS, προτείνει διαχωρισμό του περιεχομένου html από τους κανόνες παρουσίασης CSS και από των κώδικα Javascript. Έτσι, κατά την υλοποίηση της εφαρμογής μας, φροντίσαμε όλοι οι κανόνες css και όλα τα scripts Javascript να περιλαμβάνονται σε εξωτερικά αρχεία στα οποία γίνονται αναφορές από το κάθε έγγραφο html. Η παραδοσιακή HTML, στις πρώτες εκδόσεις της, βασιζόταν στο πρότυπο της SGML, η οποία είναι μια γλώσσα γενικής περιγραφής εγγράφων. Με την εξέλιξη και την διάδοση στο internet όμως, της γλώσσας XML, θεωρήθηκε αναγκαίο να υπάρξει μια εξέλιξη της γλώσσας HTML η οποία να υπακούει στους κανόνες του xml standard κληρονομώντας όλα τα θετικά χαρακτηριστικά μιας xml-based γλώσσας. Η νέα έκδοση της γλώσσας ονομάστηκε XHTML. Πλέον, τα έγγραφα xhtml, πέραν του ότι χρησιμοποιούνται από τους φυλλομετρητές για την εμφάνιση web περιεχομένου, αποτελούν έγκυρα xml έγγραφα τα οποία μπορούν να διαβαστούν και να τεθούν υπό επεξεργασία από xml parsers. Επίσης έχουν μια συγκροτημένη δομή η οποία είναι παρόμοια με όλα τα xml-based έγγραφα και η οποία επιτρέπει την ύπαρξη ενός ενοποιημένου Document Object Model (DOM) το οποίο βοηθάει στην δομική ανάλυση και επεξεργασία τέτοιων εγγράφων.

3.4 XML και Document Object Model

Επειδή πολλές από τις τεχνολογίες που χρησιμοποιούνται μέσα στην εφαρμογή μας βασίζονται στην XML (όπως η προαναφερόμενη XHTML), αλλά και επειδή σε πολλά σημεία χρησιμοποιείται καθαρά XML (όπως στην ασύγχρονη επικοινωνία με τον server και στην αποθήκευση ορισμένων αρχείων), θεωρούμε βασικό σε αυτό το σημείο να γίνει μια σύντομη περιγραφή της γλώσσας. Η XML, αναλυτικότερα η **eXtensible Markup Language,** είναι μια γλώσσα που περιλαμβάνει ένα σετ κανόνων για την κωδικοποίηση εγγράφων σε μορφή ικανή για να την «διαβάσει» ένα υπολογιστικό σύστημα. Οι σχεδιαστικοί στόχοι της XML δίνουν έμφαση στην απλότητα, γενικότητα και στην ευχρηστία μέσω internet. Η xml βασίζεται σε αρχεία κείμενου με ισχυρή υποστήριξη Unicode χαρακτήρων. Παρόλο που η XML σχεδιάστηκε με γνώμονα τα έγγραφα κειμένου, χρησιμοποιείται ευρέως και για την

περιγραφή αυθαίρετων δομών δεδομένων π.χ. web services, ή ακόμα και γραφικών όπως η SVG και η X3D. Κάθε έγγραφο XML αποτελείται από μια σειρά χαρακτήρων. Οι χαρακτήρες αυτοί χωρίζονται σε χαρακτήρες σημάνσεως και σε χαρακτήρες περιεχομένου. Η σήμανση σε ένα αρχείο xml γίνεται με στοιχεία/elements τα οποία δηλώνονται με tags (όπως ακριβώς γίνεται και στην html που περιγράψαμε στο προηγούμενο υποκεφάλαιο). Η δήλωση κάθε στοιχείου ξεκινάει με την αντίστοιχη tag ανοίγματος, ακολουθεί το περιεχόμενο και μετά η tag κλεισίματος. Τα tags μπορούν να περιέχουν χαρακτηριστικά/attributes (ίδιας μορφής σύνταξης με την html) τα οποία περιέχουν πληροφορίες για το συγκεκριμένο element. Τα elements μπορούν να είναι εμφωλευμένα όπως και στην html. Όλα τα αρχεία xml ξεκινάνε με την δήλωση:

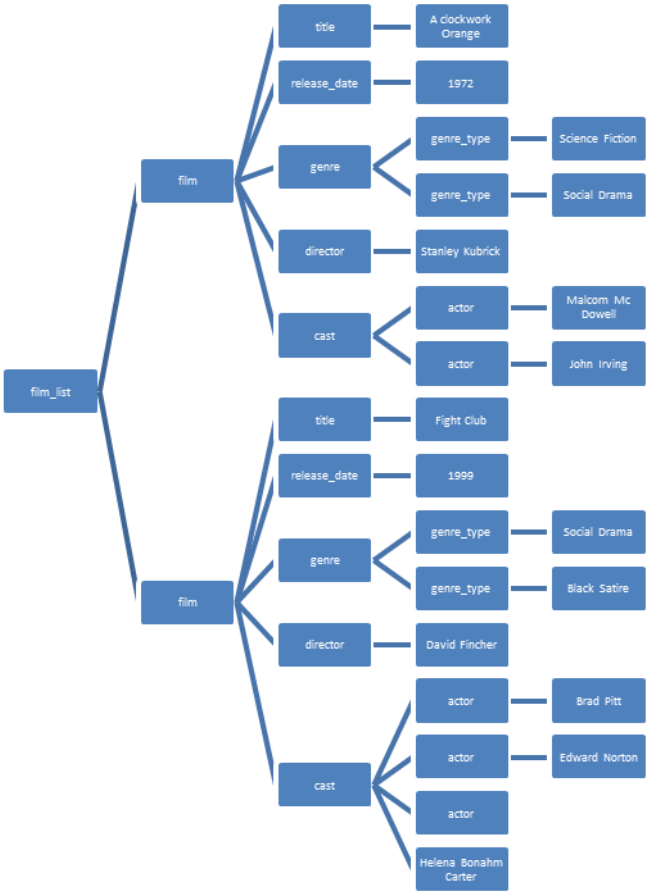
```
<?xml version="1.0" encoding="UTF-8" ?>
```

Ακολουθεί ένα παράδειγμα αρχείο XML. Έστω ότι έχουμε ένα έγγραφο που περιγράφει μια μικρή λίστα από ταινίες. Κάθε ταινία περιλαμβάνει στοιχεία όπως ο τίτλος, είδος, σκηνοθέτης, , και cast ηθοποιών. Το παραπάνω έγγραφο θα μπορούσε να περιγραφεί ως εξής σε XML:

```
<?xml version="1.0" encoding="UTF-8" ?>
<film_list>
<film>
<title lang=”en”> A clockwork Orange </title>
<genre>
<genre_type> Science Fiction </genre_type>
<genre_type> Social Drama </genre_type>
</genre>
<release_date> 1972 </release_date>
<director nationality=”American”> Stanley Kubrick </director>
<cast>
<actor> Malcom McDowell </actor>
<actor> John Clive </actor>
</cast>
</film>
<film>
<title lang=”en”> Fight Club </title>
<genre>
<genre_type> Social Drama </genre_type>
```

```
<genre_type> Black Satire </genre_type>
</genre>
<release_date> 1999 </release_date>
<director nationality=”American” > David Fincher </director>
<cast>
<actor> Brad Pitt </actor>
<actor> Edward Norton </actor>
<actor> Helena Bonahm Carter </actor>
<cast>
</film>
</film_list>
```

Παρατηρούμε ότι το root element του εγγράφου είναι η film_list, η οποία περιλαμβάνει εμφωλευμένα δύο άλλα elements τύπου film κτλ. Παρατηρώντας την δομή του αρχείου, με τον τρόπο που είναι εμφωλευμένα τα στοιχεία το ένα μέσα στο άλλο, δημιουργούν μια νοητή δενδρική δομή η οποία θα είχε την εξής μορφή:



Εικόνα 33 – Δενδρική δομή του XML DOM ενός XML αρχείου

Αυτή η δενδρική αναπαράσταση εγγράφων XML, αλλά και γενικά εγγράφων τα οποία περιγράφονται από γλώσσες mark-up που αποτελούν επεκτάσεις της γλώσσας XML, ονομάζεται Document Object Model και αποτελεί μια δενδρική δομή δεδομένων, η οποία δημιουργείται αναλύοντας την δομή του εγγράφου. Η δομή αυτή συμπεριφέρεται σε κάθε στοιχείο (element) του εγγράφου σαν να είναι αντικείμενο/object. Παραδείγματος χάριν, στην περίπτωση ιστοσελίδων σε μορφή XHTML, ο φυλλομετρητής αναλαμβάνει την διαδικασία να αναλύσει την XML μορφή της ιστοσελίδας και παράγει την αντίστοιχη δενδρική μορφή των html στοιχείων της σελίδας σε ιεραρχία. Το Document Object Model, αν και καταλαμβάνει αρκετούς πόρους για να υλοποιηθεί, είναι απαραίτητο για την πρόσβαση από γλώσσες προγραμματισμού (όπως η Javascript) στα αντικείμενα/στοιχεία της σελίδας. Ο φυλλομετρητής παράγει αναφορά στο root στοιχείο του Dom και από εκεί και πέρα υπάρχουν συναρτήσεις για επιτευχθεί διάσχιση όλου το δέντρο κόμβο – κόμβο. Κάθε κόμβος του δέντρου αντιπροσωπεύει ένα στοιχείο της html σελίδας. Το στοιχείο αυτό συμπεριφέρεται σαν object μέσα στα πλαίσια του Dom και έχει της εξής ιδιότητες:

- **Element.innerHTML** επιστρέφει το HTML περιεχόμενο του στοιχείου element.
- **Element.nodeName** επιστρέφει το όνομα του στοιχείου element.

- **Element.nodeValue** επιστρέφει την τιμή του στοιχείου element.
- **Element.parentNode** επιστρέφει τον γονέα του αντικειμένου element.
- **Element.childNodes** επιστρέφει λίστα με τα αντικείμενα παιδιά του element.
- **Element.attributes** επιστρέφει λίστα με τα attributes του στοιχείου element.

Επειδή όμως τις περισσότερες φορές η διάσχιση μιας ολόκληρης δενδρικής δομής μέχρι να βρούμε το στοιχείο που μας ενδιαφέρει είναι χρονοβόρα διαδικασία μας παρέχονται και οι εξής χρήσιμες συναρτήσεις

- **getElementsByTagName(name)** είναι μια συνάρτηση που δέχεται το όνομα μιας tag και επιστρέφει λίστα με όλα τα αντικείμενα/στοιχεία που περιγράφονται από την συγκεκριμένη tag, π.χ. η **getElementsByTagName(p)** επιστρέφει μια λίστα με όλα τα στοιχεία <p> της σελίδας που είναι παράγραφοι κειμένου.
- **getElementById(name)** είναι μια συνάρτηση που επιστρέφει ένα συγκεκριμένο element της σελίδας που έχει το σαν τιμή στο attribute id το "name". Η **getElementById()** χρησιμοποιείται ευρέως για να επιστραφεί αναφορά σε ένα συγκεκριμένο element της σελίδας, ώστε να εφαρμοστεί σε αυτό π.χ. ένας κανόνας CSS ή να δεχθεί επεξεργασία από κώδικα Javascript.

Το Document Object Model είναι απαραίτητο για την δημιουργία δυναμικών σελίδων ή για την επεξεργασία μέσω scripts εγγράφων που βασίζονται στο πρότυπο xml. Παρόμοιοι κανόνες και συναρτήσεις που αναφέρθηκαν παραπάνω ισχύουν και σε έγγραφα SVG αλλά και X3D που θα εξετάσουμε αργότερα.

3.5 Javascript

Η Javascript αποτελεί scripting γλώσσα η οποία εκτελείται στο περιβάλλον του φυλλομετρητή και μπορεί να διαχειριστεί και να μεταβάλει τα html στοιχεία της ιστοσελίδας, συμπεριφέροντας τα σαν αντικείμενα/objects. Αν και ξεκίνησε αρχικά σαν μια απλή scripting γλώσσα για την προσθήκη μικρών δυναμικών λειτουργιών στις ιστοσελίδες έχει εξελιχθεί σε μια ισχυρή γλώσσα προγραμματισμού με υποστήριξη δυναμικών συναρτήσεων, δομών οντοκεντρικού προγραμματισμού, μεθόδους που εκτελούνται ταυτόχρονα (σαν ψευδό-threads), ασύγχρονη επικοινωνία με τον server μέσω XML. Η Javascript εκτελείται δυναμικά, αυτό σημαίνει ότι ο κώδικας δεν μεταγλωττίζεται στο σύνολο του πριν την έναρξη της εκτέλεσης, αλλά διερμηνεύεται εντολή προς εντολή σε πραγματικό χρόνο. Αυτό επιτρέπει ιδιαίτερες τεχνικές δυναμικού προγραμματισμού, που δεν τις συναντούμε σε άλλες παραδοσιακές γλώσσες, αλλά ταυτόχρονα έχουμε επιβάρυνση στην ταχύτητα εκτέλεσης. Ωστόσο, με την εξέλιξη των υπολογιστικών συστημάτων, αλλά και των φυλλομετρητών έχουν εμφανιστεί μηχανές εκτέλεσης της Javascript (όπως π.χ. η V8

engine του Google Chrome), οι οποίες επιτυγχάνουν όλο και μεγαλύτερες ταχύτητες εκτέλεσης. Αυτό έχει ως αποτέλεσμα τα προγράμματα Javascript στις ιστοσελίδες να γίνονται όλο και πιο περίπλοκα, με δυνατότητες ανάλογες προγραμμάτων desktop εφαρμογών.

Μερικά από τα χαρακτηριστικά της γλώσσας περιγράφονται παρακάτω:

3.5.1 Δομημένη Σύνταξη

Η σύνταξη της Javascript μοιάζει με την σύνταξη της C ή της JAVA. Ο κώδικας δομείται μέσα σε blocks τα οποία ορίζονται από τις αγκύλες { και }. Επιτρέπονται statements όπως while, for, if και switch. Τα ερωτηματικά στο τέλος των εντολών μπορούν να παραλείπονται.

3.5.2 Δυναμικός χαρακτήρας

Η Javascript επιτρέπει δυναμικούς τύπους μεταβλητών. Αυτό σημαίνει ότι μια μεταβλητή, π.χ. x, μπορεί αρχικά να πάρει την τιμή 5, οπότε να αντιπροσωπεύει integer, αλλά αργότερα μπορούμε να της εκχωρήσουμε σαν τιμή ένα αλφαριθμητικό (π.χ. x = “sun”) και αυτόματα να μετατραπεί σε μεταβλητή τύπου string. Επίσης περιλαμβάνει την συνάρτηση eval, η οποία επιτρέπει την δυναμική εκτέλεση κώδικα ο οποίος δίδεται σαν αλφαριθμητικό σε run-time.

3.5.3 Αντικείμενα

Τα πάντα στην Javascript αποτελούν αντικείμενα/objects. Τα αντικείμενα της Javascript είναι στην ουσία πίνακες συσχετισμού. Παραδείγματός χάριν, έστω ότι έχουμε ένα αντικείμενο που περιγράφει την οντότητα «πρόσωπο» και έχει της εξής ιδιότητες: όνομα, επώνυμο και ηλικία. Σε μια κλασσική οντοκεντρική γλώσσα, το αντικείμενο «πρόσωπο» θα επέτρεπε προσπέλαση στις ιδιότητές του ως εξής:

```
Person.name = “Bob”

Person.surname = “Dylan”

Person.age = 69;
```

Με τον ίδιο τρόπο επιτρέπει και η Javascript την πρόσβαση στις ιδιότητες ενός αντικειμένου, μόνο που εφόσον αποτελεί πίνακα συσχετισμού η πρόσβαση μπορεί να γίνει και ως εξής:

```
Person[“name”] = “Bob”;

Person[“surname”] = “Dylan”;
```

```
Person[“age”] = 69;
```

Βλέπουμε ότι τα ονόματα των ιδιοτήτων αποτελούν κλειδιά αλφαριθμητικών στον πίνακα συσχετισμού. Κατά την εκτέλεση του προγράμματος είναι δυνατή η δυναμική προσθήκη και διαγραφή ιδιοτήτων σε ένα αντικείμενο.

3.5.4 Συναρτήσεις

Οι συναρτήσεις στην Javascript είναι τύπου first class functions, οι οποίες θεωρούνται αντικείμενα. Εκτός από σώμα εκτελέσιμου κώδικα, ο οποίος μπορεί να καλεστεί ανά πάσα στιγμή, έχουν ιδιότητες όπως κάθε άλλο αντικείμενο στην Javascript. Επίσης επιτρέπεται η δήλωση συναρτήσεων μέσα σε σώματα άλλων συναρτήσεων (inner functions). Οι συναρτήσεις αυτές έχουν πρόσβαση στις μεταβλητές τις εξωτερικής συνάρτησης ακόμα και αν αυτή έχει τελειώσει την εκτέλεσή της. Αυτός είναι ο μηχανισμός με τον οποίο δημιουργούνται τα λεγόμενα closures στην Javascript.

3.5.5 Prototypes

Τα prototypes είναι ο μηχανισμός που χρησιμοποιεί η Javascript αντί των κλάσεων και της κληρονομικότητας. Μπορούμε να εξομοιώσουμε πολλά στοιχεία του οντοκεντρικού προγραμματισμού χρησιμοποιώντας prototypes στην Javascript. Για object constructors χρησιμοποιούνται απλές συναρτήσεις οι οποίες καλούνται με την λέξη κλειδί new, σαν πρόθεμα, ώστε να δημιουργήσουν ένα νέο αντικείμενο. Η λέξη κλειδί this, στο σώμα μιας συνάρτησης η οποία καλείται σαν object constructor, αναφέρεται στο καινούριο αντικείμενο που δημιουργήθηκε. Για να αντιληφθούμε τον μηχανισμό δημιουργίας αντικειμένων στην Javascript παραθέτουμε το παρακάτω παράδειγμα:

Έστω να θέλουμε να δημιουργήσουμε ένα αντικείμενο που αναπαριστά το γεωμετρικό σχήμα παραλληλόγραμμο, αποθηκεύοντας πληροφορίες για το πλάτος, μήκος και έχοντας μια μέθοδο που να υπολογίζει το εμβαδό της επιφάνειάς του. Έστω το όνομα τις κλάσης του αντικειμένου είναι Rectangle. Σε Javascript η δήλωση της κλάσης μπορεί να γίνει ως εξής:

```
Function Rectangle(w,h){

this.width = w;

this.height = h;

}
```

Η παραπάνω συνάρτηση δουλεύει ως object constructor. Δηλαδή όταν θα θέλουμε να δημιουργήσουμε ένα νέο παραλληλόγραμμο θα καλούμε π.χ. `myRectangle =`

`new Rectangle(10,20)` και θα δημιουργείται ένα object τύπου rectangle, με τιμές στο property width = 10 και στο property height = 20. Αν θέλουμε να προσθέσουμε την μέθοδο getArea() τότε θα πρέπει να χρησιμοποιήσουμε τα prototypes. Η δήλωση γίνεται ως εξής:

```
Rectangle.prototype.getArea = function()

{

Return this.width * this.height;

}
```

Η κλήση της στο αντικείμενο myRectangle γίνεται ώς εξής:

```
myRectangle.getArea() // επιστρέφει 200
```

Γενικά αντιλαμβανόμαστε ότι τα αντικείμενα στην Javascript περιλαμβάνουν ιδιότητες/properties οι οποίες έχουν συγκεκριμένες τιμές, για το κάθε αντικείμενο ξεχωριστά, αλλά μπορούν να περιέχουν και properties οι οποίες είναι ίδιες για όλα τα αντικείμενα μιας συγκεκριμένης κλάσης (όπως π.χ. οι μεταβλητές static στην java). Οι properties οι οποίες παραμένουν ίδιες για όλα τα αντικείμενα της Javascript, δηλώνονται στο prototype μέρος ενός αντικειμένου. Για αυτό και οι μέθοδοι, οι οποίες στην ουσία αποτελούν μια συνάρτηση που εκτελείται με τον ίδιο τρόπο για κάθε αντικείμενο της κλάσης, δηλώνονται σαν functions στο prototype μέρος του αντικειμένου. Είναι καλή τεχνική να διαχωρίζεται ο κώδικας της Javascript από το σώμα της html σελίδας και να περιέχεται σε εξωτερικά αρχεία script. Τα αρχεία αυτά γίνονται include στην html σελίδα χρησιμοποιώντας το tag script:

```
<script type="text/Javascript" src="…/scripts/script1.js"></script>
```

Για να «στοχεύσουμε» στοιχεία της html σελίδας μέσα στον κώδικα Javascript και να τους δώσουμε interactivity, θα πρέπει τα στοιχεία αυτά να έχουν το attribute id ίσον με μια ονομασία η οποία είναι μοναδική σε όλο το έγγραφο. Μετά, μέσα στον κώδικα Javascript μπορεί να γίνει αναφορά, στο συγκεκριμένο στοιχείο, εκτελώντας την εντολή document.getItemById(name). Παραδείγματος χάριν, έστω ότι θέλουμε να αλλάξουμε το κείμενο μιας παραγράφου στην οποία έχουμε δώσει το διακριτικό id = "main_paragraph". Η δήλωση στο tag της παραγράφου θα γίνει ως εξής:

```
<p id= "main_paragraph"> …. Paragraph Content </p>
```

Και η αναφορά στο στοιχείο της παραγράφου μέσα στον κώδικα Javascript γίνεται ως εξής:

```
var paragraph = document.getElementById("main_paragraph");

paragraph.innerHtml = "new paragraph content…";
```

Σημειώνουμε ότι η δήλωση μεταβλητών στην Javascript γίνεται με το keyword var το οποίο είναι προαιρετικό. Επίσης, η Javascript υποστηρίζει ένα σύστημα events για την προσθήκη διαδραστικότητας στα στοιχεία μιας ιστοσελίδας. Κάθε στοιχείο ιστοσελίδας περιλαμβάνει κάποια events τα οποία μπορούν να ενεργοποιήσουν την εκτέλεση κώδικα Javascript. Π.χ. μπορούμε να χρησιμοποιήσουμε το event onClick ενός στοιχείου εικόνας html ώστε να εκτελείται ένα κομμάτι κώδικα όταν ο χρήστης κάνει κλικ πάνω στην εικόνα. Μερικά από τα events που υποστηρίζονται στην Javascript, όσον αφορά τα στοιχεία μιας html σελίδας, είναι τα εξής:

- onclick – Όταν ο χρήστης κλικάρει ένα στοιχείο html.
- ondblclick – Όταν ο χρήστης κάνει διπλό κλικ σε ένα στοιχείο html.
- onmousemove – Όταν το ποντίκι κινείται.
- onmouseover – Όταν το ποντίκι εισέρχεται στην περιοχή ενός στοιχείου.
- onmouseout – Όταν το ποντίκι εξέρχεται από την περιοχή ενός στοιχείου.
- onload – Όταν τελειώνει το φόρτωμα μιας σελίδας.
- onunload – Όταν ο χρήστης βγαίνει από μια ιστοσελίδα.
- onmousedown – Όταν πατιέται το πλήκτρο του ποντικιού.
- onmouseup – Όταν ελευθερώνεται το πλήκτρο του ποντικιού.
- onchange – Όταν αλλάζει η κατάσταση ενός αντικειμένου html π.χ. όταν αλλάζει το κείμενο σε ένα πλαίσιο κειμένου.
- onkeypress – Όταν πατιέται ένα πλήκτρο από το πληκτρολόγιο.

Στην Javascript ενσωματώνουμε στα στοιχεία της html, event listeners ώστε να «πιάνουν» (capture) τα events που δημιουργούνται πάνω στο συγκεκριμένο στοιχείο. Αν όμως δεν έχουμε επισυνάψει έναν event listener σε ένα στοιχείο και ο χρήστης αλληλεπιδράσει μαζί του (π.χ. κάνει κλικ με το ποντίκι) τότε ο φυλλομετρητής θα κοιτάξει στην ιεραρχία των στοιχείων html, να βρει πιο στοιχείο-γονέας έχει προσαρμοσμένο έναν event-listener, ώστε να δεσμεύσει και να διαχειριστεί το event. Έτσι στην Javascript τα events αναφέρεται ότι έχουν μορφή φυσαλίδας, καθώς ανεβαίνουν προς τα πάνω στην ιεραρχία των στοιχείων μέχρι να βρεθεί στον δρόμο τους ένας κατάλληλος event listener να τα διαχειριστεί. Κάθε event στην Javascript αποτελεί αντικείμενο το οποίο μπορεί να έχει τις εξής ιδιότητες

- type – περιέχει το είδος του event.
- timestamp – η ώρα που συνέβη το event.
- target – το στοιχείο που ενεργοποίησε το event.

- currentTarget – το στοιχείο που δέσμευσε το event καθώς ανέβαινε σε ιεραρχία.
- clientX, clientY – συντεταγμένες x,y του κέρσορα του ποντικιού στα events που σχετίζονται με το ποντίκι.

3.6 Cascading Style Sheets

Τα Cascading Style Sheets είναι μια γλώσσα περιγραφής κανόνων που διέπουν την παρουσίαση και την μορφή των στοιχείων ενός εγγράφου που περιγράφεται από μια γλώσσα επισήμανσης όπως η HTML ή η XML. Κάθε κανόνας CSS, στοχεύει ένα συγκεκριμένο στοιχείο (με βάση το id του) ή μια ομάδα στοιχείων με βάση την κλάση τους ή με βάση το tag τους. Συγκεκριμένα, στην html οι κανόνες css μπορούν να είναι ενσωματωμένοι μέσα στο αρχείο της ιστοσελίδας ανάμεσα στα tags <style> … </style>. Συνίσταται όμως η διαχώριση του περιεχομένου από τις λεπτομέρειες παρουσίασής του, οπότε θεωρείται καλή τεχνική οι κανόνες css να βρίσκονται σε εξωτερικά αρχεία (με κατάληξη *.css), τα οποία μπορούν να συνδεθούν με το έγγραφο html χρησιμοποιώντας το tag <link src=”…/css/style1.css” type=”text/css”>…</link>. Παρακάτω δίδονται κάποια παραδείγματα για να εξεταστούν οι διάφορες μορφές που μπορούν να έχουν οι κανόνες css. Π.χ. έστω ότι θέλουμε να καθορίσουμε την γραμματοσειρά και το μέγεθος κειμένου μιας παραγράφου με id = “paragraph1”. Ο κανόνας css θα είναι ο εξής:

```
#paragraph1

{

font-family : “Times New Roman”,

font-size : 10px

}
```

Παρατηρούμε ότι ο κανόνας ξεκινάει με την αναφορά του αντικειμένου που θέλουμε να επεξεργαστούμε και μετά σε ένα block αγκύλων ακολουθούν οι κανόνες τις εμφάνισής του που θέλουμε να καθορίσουμε. Όταν στοχεύουμε ένα αντικείμενο με βάση το id τοποθετούμε το σύμβολο “#” πριν από το id. Έστω ότι θέλουμε να αλλάξουμε την στοίχιση του κειμένου σε όλα τα αντικείμενα της κλάσης “class1”. Ο κανόνας css θα είναι ο ακόλουθος:

```
.class1 {

Text-align: center

}
```

Βλέπουμε πως το όνομα της κλάσης θα πρέπει μπροστά του να έχει το σύμβολο “.”

Έστω ότι θέλουμε να αλλάξουμε το στην στοίχιση κειμένου στα στοιχεία <p> ενός εγγράφου html τότε ο κανόνας θα είναι ο εξής:

```
P {

Text-align : center

}
```

Επίσης μπορούμε να δηλώσουμε συγκεκριμένα αντικείμενα μιας κλάσης να υπακούσουν σε έναν κανόνα, π.χ. θέλουμε όλες οι επικεφαλίδες 1 (h1) της κλάσης = "attention" να έχουν χρώμα κειμένου κόκκινο, Ο κανόνας που πρέπει να εφαρμοστεί είναι ο εξής:

```
H1.attention {

color: red

}
```

3.7 SVG

Η SVG είναι μια γλώσσα, βασισμένη στην XML, η οποία χρησιμοποιείται για την περιγραφή δισδιάστατων διανυσματικών γραφικών. Υποστηρίζεται εγγενώς από πολλούς φυλλομετρητές. Τα διανυσματικά γραφικά που περιγράφονται από αρχεία SVG δεν χάνουν την ποιότητά τους όταν τα μεγεθύνουμε ή όταν κάνουμε zoom σε αυτά. Κάθε γραφικό σε ένα αρχείο SVG αποτελεί ένα στοιχείο element με δικές του ιδιότητες και με μια δικιά του θέση στην δενδρική δομή SVG DOM. Όλες αυτές οι ιδιότητες ενός στοιχείου SVG μπορούν να αλλαχθούν χρησιμοποιώντας Javascript. Τα γραφικά SVG μπορούν να δηλωθούν σε εξωτερικά αρχεία (με κατάληξη *.SVG) ή να βρίσκονται ενσωματωμένα μέσα σε σελίδες html χρησιμοποιώντας το tag <SVG> … SVG content … </SVG>. Το κεντρικό root element σε ένα αρχείο γραφικών SVG είναι το SVG element, το οποίο στην ουσία καθορίζει τον καμβά σχεδίασης των διανυσματικών γραφικών. Μέσα στο SVG element υπάρχουν εμφωλευμένα στοιχεία γραφικών τα οποία περιγράφονται με tags. Π.χ. ένα τετράγωνο σχήμα σε γλώσσα SVG περιγράφεται ως εξής:

```
<rect width="300" height="300" x="20" y="20" fill="red" stroke="black" stroke-width="2" />
```

Αναλύοντας την παραπάνω δήλωση, βλέπουμε πως η SVG «ζητάει» από τον browser να σχεδιάσει ένα παραλληλόγραμμο σχήμα με ύψος/πλάτος = 300 pixels (άρα είναι τετράγωνο), να το τοποθετήσει στις συντεταγμένες 20,20 του καμβά σχεδίασης, να το γεμίσει με κόκκινο χρώμα στο εσωτερικό και η γραμμή του περιγράμματος να είναι χρώματος μαύρου και πάχους 2 pixels. Παρομοίως θα μπορούσε να σχεδιαστεί ένας κύκλος, με ίδιο χρώμα γεμίσματος και ίδια γραμμή περιγράμματος, αλλά διαφορετικές διαστάσεις, χρησιμοποιώντας το αντίστοιχο tag:

```
<circle cx=”20” cy=”20” r=”150” fill=”red” stroke=”black” stroke-width=”2” />
```

Τα σχήματα μπορούν να ομαδοποιηθούν σε groups χρησιμοποιώντας το tag <g>. Π.χ. τα παραπάνω σχήματα που περιγράφηκαν μπορούν να ομαδοποιηθούν σε ένα group ως εξής:

```
<g><rect width=”300” height=”300” x=”20” y=”20” fill=”red” stroke=”black” stroke-width=”2” /><circle cx=”20” cy=”20” r=”150” fill=”red” stroke=”black” stroke-width=”2” /></g>
```

Επίσης μπορούν να δοθούν μετασχηματισμοί στα σχήματα όπως μεταθέσεις, περιστροφές, κλιμάκωση κτλ χρησιμοποιώντας σε κάθε στοιχείο το attribute transform. Π.χ. αν θέλουμε να μεταθέσουμε το τετράγωνο 100 pixels κατά τον άξονα x και 50 pixels κατά τον άξονα y θα γράφαμε:

```
<rect width=”300” height=”300” x=”20” y=”20” fill=”red” stroke=”black” stroke-width=”2” transform=”translate(100,50) />
```

Για περιστροφή π.χ. 20 μοιρών θα δηλώναμε:

```
<rect width=”300” height=”300” x=”20” y=”20” fill=”red” stroke=”black” stroke-width=”2” transform=”rotate(20) />
```

Οι μετασχηματισμοί των σχημάτων μπορούν να δηλωθούν με την μορφή πίνακα μετασχηματισμού 3x3 όπως ορίζει η γραμμική άλγεβρα:

a	B	c
d	e	f
0	0	1

Η παράμετροι του πίνακα, που μπορούν να χρησιμοποιηθούν για να εφαρμοστούν μετασχηματισμοί στην δισδιάστατη γεωμετρία των σχημάτων SVG, είναι οι a, b, c, d, e, f οι οποίοι συνήθως εκφράζονται και σαν διάνυσμα [a b c d e f]. Οι μετασχηματισμοί εκφράζονται μέσω του πίνακα ως εξής:

- Μετατόπιση tx κατά τον άξονα x και ty κατά τον άξονα y:

1	0	tx
0	1	ty
0	0	1

- Περιστροφή κατά γωνία a:

Cos(a)	-sin(a)	0
Sin(a)	Cos(a)	0
0	0	1

- κλιμάκωση κατά sx και sy:

sx	0	0
1	sy	0
0	0	1

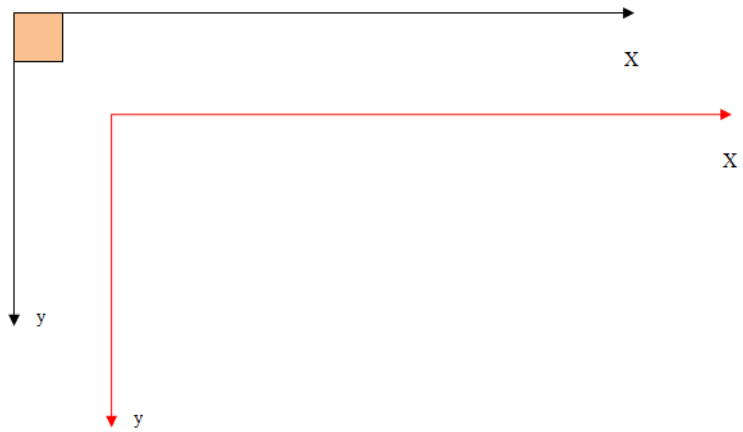
Ο καμβάς σχεδίασης, στο σχέδιο SVG, αντιπροσωπεύεται από το SVG Element <SVG> και ορίζει το αρχικό σύστημα συντεταγμένων το οποίο είναι το εξής:

Εικόνα 34 – Αρχικό σύστημα συντεταγμένων μιας περιοχής σχεδίασης SVG

Όταν εφαρμόζουμε μετασχηματισμούς στα στοιχεία SVG αλλάζουμε σύστημα συντεταγμένων. Ειδικά όταν εφαρμόζουμε μετασχηματισμούς σε στοιχεία <g>, τα οποία περιέχουν μέσα τους μια ομάδα από σχήματα, μεταπηδάμε σε ένα ξεχωριστό σύστημα συντεταγμένων για το συγκεκριμένο γκρουπ σχημάτων. Παραδείγματος χάριν, έστω ότι έχουμε στον καμβά ένα τετράγωνο το οποίο βρίσκεται στο σημείο 0,0 με μέγεθος 10 (Εικόνα 35).

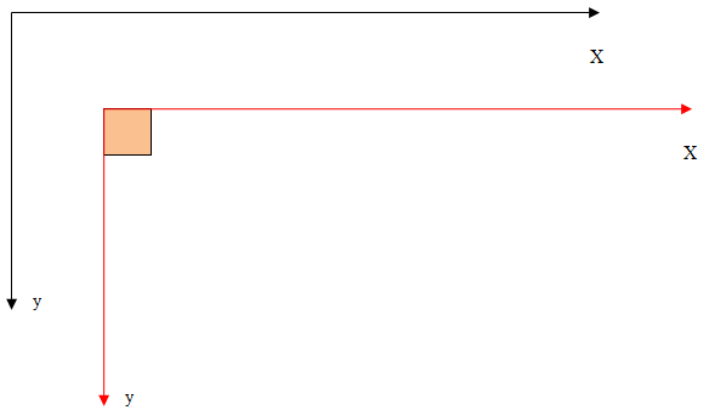
Εικόνα 35 – Αρχικό σύστημα συντεταγμένων με στοιχείο <rect> τοποθετημένο στο σημείο (0,0)

Έστω ότι δηλώνουμε ένα στοιχείο <g>, το οποίο δεν περιλαμβάνει ακόμα μέσα του κανένα γεωμετρικό σχήμα. Εφαρμόζουμε στο στοιχείο g έναν μετασχηματισμό για μετατόπιση 20 κατά άξονα x και 20 κατά άξονα y. Αυτόματα δημιουργείται ένα νέο σύστημα συντεταγμένων αποκλειστικά για αυτό το group (Εικόνα 36). Στο σχήμα σημειώνεται με κόκκινο το νέο σύστημα συντεταγμένων του στοιχείου <g>.



Εικόνα 36 – Σύστημα συντεταγμένων του <g>

Αν, χωρίς να αλλάξουμε τις ιδιότητες του αντικειμένου <rect>, το μετακινήσουμε μέσα στο στοιχείο <g>, τότε παρόλο που το τετράγωνο περιγράφει ότι βρίσκετε στο (0,0) του άξονα των συντεταγμένων, επειδή τώρα υπακούει στο σύστημα συντεταγμένων του στοιχείου <g> θα βρίσκεται στο (0,0) του κόκκινου άξονα όπως φαίνεται στην παρακάτω εικόνα:



Εικόνα 37 – Τοποθέτηση του στοιχείου <rect> μέσα στο <g>

Επειδή όμως ο κόκκινος άξονας είναι ήδη μετατοπισμένος λόγω του μετασχηματισμού που έχουμε επιβάλει στο στοιχείο <g>, το τετράγωνο τελικά βρίσκεται στο σημείο 20,20 όσον αφορά το αρχικό σύστημα συντεταγμένων του καμβά σχεδίασης. Στα αρχεία SVG έχουμε πολλές ομαδοποιήσεις γραφικών χρησιμοποιώντας στοιχεία <g> στα οποία επιβάλλουμε μετασχηματισμούς και δημιουργούνται νέα συστήματα συντεταγμένων, πολλές φορές εμφωλευμένα το ένα μέσα στο άλλο. Στο προηγούμενο παράδειγμα αν εξετάζαμε τις x και y ιδιότητες του στοιχείου <rect> θα παίρναμε τις συντεταγμένες 0,0. Αλλά επειδή είναι τοποθετημένο μέσα στο group του στοιχείου <g> στο οποίο εφαρμόζεται ένας μετασχηματισμός μετατόπισης κατά 20,20 αν εφαρμόσουμε τον μετασχηματισμό με την μορφή πίνακα στις συντεταγμένες του <rect> θα πάρουμε το αποτέλεσμα 20,20 το οποίο είναι και σωστό.

Η ιεραρχία των γραφικών στοιχείων μέσα σε ένα αρχείο SVG δημιουργεί μια δενδρική δομή (όπως και στα αρχεία HTML) SVG Document Object Model στην οποία τα SVG elements αναπαριστούνται ως objects με ιδιότητες τις οποίες μπορούμε να αλλάξουμε μέσω τις Javascript. Η Javascript μπορεί να βρίσκεται μέσα στο σώμα ενός SVG εγγράφου ή σε εξωτερικό αρχείο.

Όταν εξετάζουμε ένα αντικείμενο γραφικών, το οποίο βρίσκεται εμφωλευμένο μέσα σε ένα άλλο σύστημα συντεταγμένων, και θέλουμε να λάβουμε τις συντεταγμένες του ως προς το αρχικό σύστημα συντεταγμένων του καμβά, μπορούμε να χρησιμοποιήσουμε την getCTM() μέθοδο. Η μέθοδος αυτή επιστρέφει τον Current Transformation Matrix, δηλαδή τον πίνακα όλων των εμφωλευμένων μετασχηματισμών που έχει υποστεί το στοιχείο αυτό.

Πέρα από τα στοιχεία SVG τα οποία αποτελούν σχήματα που εμφανίζονται στην οθόνη υπάρχουν και άλλα στοιχεία τα οποία δεν αναπαριστούνται γραφικά (non-renderable) αλλά έχουν βοηθητικό ρόλο:

- **SVGPoint** είναι ένα στοιχείο το οποίο αναπαριστά ένα σημείο στο επίπεδο και έχει σαν ιδιότητες τις συντεταγμένες του σημείου αυτού.

- **SVGMatrix** είναι ένα στοιχείο που αναπαριστά έναν πίνακα μετασχηματισμού. Έχει διάφορες ιδιότητες και μεθόδους όπως ο υπολογισμός του αντίστροφου πίνακα.

3.7 X3D

Η X3D τεχνολογία αποτελεί εξέλιξη της γλώσσας VRML, για την περιγραφή τρισδιάστατων γραφικών σε φυλλομετρητές. Βασίζεται στην γλώσσα XML και χρησιμοποιεί tags για να περιγράψει βασικά γεωμετρικά σχήματα όπως ο κύβος, ο κύλινδρος, το επίπεδο κτλ.

Το κεντρικό στοιχείο του X3D αρχείου είναι το element <scene> το οποίο περιλαμβάνει μέσα του εμφωλευμένα όλα τα γεωμετρικά στοιχεία της σκηνής. Κάθε τρισδιάστατο σχήμα περιγράφεται από το tag <shape>. Μέσα στο tag <shape> υπάρχει το tag <Appearance> το οποίο περιγράφει την εμφάνιση του τρισδιάστατου αντικειμένου. Μέσα στο tag <Appearance> τοποθετείται εμφωλευμένο είτε το tag <Material> το οποίο περιγράφει το υλικό (χρώμα/σκίαση) το οποίο εφαρμόζεται στο σχήμα, ή το <ImageTexture> αν χρησιμοποιούμε μια δισδιάστατη εικόνα σαν υφή πάνω στο αντικείμενο. Κλείνοντας το tag Appearance ακολουθεί το tag που περιγράφει την γεωμετρία του σχήματος, π.χ. αν πρόκειται για κύβο το αντίστοιχο tag είναι το <box> με attribute to size το οποίο παίρνει ένα διάνυσμα τριών τιμών που περιγράφουν τις διαστάσεις του σχήματος. Στην περίπτωση του κύβου και οι τρείς τιμές είναι ίδιες δηλ:

```
<box size=”5 5 5” />
```

Στην περίπτωση σφαίρας θα χρησιμοποιούσαμε το tag:

```
<sphere radius=”5” />
```

Το root στοιχείο κάθε εγγράφου X3D είναι το X3D element:

```
<X3D> … scene content </X3D>
```

Επίσης περιλαμβάνονται tags τα οποία περιγράφουν τον τρόπο με τον οποίο ο χρήστης αλληλεπιδρά μέσα στην σκηνή όπως το <Navigation Info> στις properties του οποίου καθορίζουμε το ύψος του avatar του χρήστη, τον τρόπο που περιηγείται στην σκηνή (WALK, FLY, EXAMINE) κτλ.

Αν θέλουμε να περιγράψουμε πιο περίπλοκη γεωμετρία χρησιμοποιούμε το tag IndexedFaceset το οποίο περιλαμβάνει μια λίστα από Vertices και πώς συνδέονται μεταξύ τους ώστε να δημιουργήσουν , edges, faces και εν τέλει ένα τρισδιάστατο μοντέλο.

Όπως κάθε έγγραφο το οποίο βασίζεται σε XML έτσι και η X3D περιλαμβάνει X3D document object model με δυνατότητα scripting μέσω της Javascript.

3.8 PHP και MYSQL

Η PHP όπως αναφέρθηκε, αποτελεί μια δυναμική γλώσσα προγραμματισμού, η οποία όμως εκτελείται μόνο σε περιβάλλον server (όπως η asp). Τα αρχεία PHP, περιλαμβάνουν κώδικα PHP σε συνδυασμό με κώδικα html και Javascript. Ο server αφού εκτελέσει τον κώδικα PHP, επιστρέφει το έγγραφο σε μορφή html, με το περιεχόμενο να έχει καθοριστεί δυναμικά. Η PHP επιτρέπει σύνδεση με έναν μεγάλο αριθμό βάσεων δεδομένων αλλά συνήθως στα web projects η PHP χρησιμοποιείται με την MySQL. Η MySQL αποτελεί έναν server βάσεων δεδομένων ο οποίος είναι κατάλληλος τόσο για μικρά όσο και για μεγάλα projects. Η MySQL υποστηρίζει scripts σε sql για τον έλεγχο της βάσης δεδομένων.

Η PHP σαν γλώσσα προγραμματισμού μοιάζει στην C. Οι δηλώσεις μεταβλητών γίνονται πάντα με το πρόθεμα $ μπροστά από το όνομα της μεταβλητής. Επίσης η PHP δεν απαιτεί δήλωση τύπων στης μεταβλητές. Ο κώδικας PHP πάντα βρίσκεται ανάμεσα από τα tags <?PHP … ?>.Υποστηρίζονται οι δομές ελέγχου if, while, for, switch κτλ όπως και στην C καθώς επίσης έχουμε και υποστήριξη συναρτήσεων.

Η MySql σαν βάση δεδομένων αποτελείται από πίνακες που περιέχουν δεδομένα σε μορφή στηλών. Παραδείγματος χάριν θα μπορούσαμε σε μια βάση δεδομένων να έχουμε έναν πίνακα με την ονομασία “customers” (πελάτες) :

id	Name	Surname	Tel	City
1	John	King	555 23423	Dallas
2	John	Delbe	222 34345	New York
3	Nick	Nate	222 34355	New York
4	Dave	York	111 33433	Detroit

Μπορούμε να εκτελέσουμε ερωτήματα σε μορφή κώδικα sql για να ανακτήσουμε δεδομένα από τον συγκεκριμένο πίνακα. Έστω ότι θέλουμε πληροφορίες για εκείνους τους πελάτες που ζουν στην New York. Το sql script θα πάει ως εξής

```
SELECT * FROM Customers WHERE City = “New York”
```

Τα αποτελέσματα θα είναι τα εξής:

id	Name	Surname	Tel	City

2	John	Delbe	222 34345	New York
3	Nick	Nate	222 34355	New York

Βλέπουμε ότι τα ερωτήματα στην απλή μορφή τους έχουν τρία τμήματα

- SELECT: ακολουθούν οι στήλες του πίνακα οι οποίες θέλουμε να εμφανιστούν στο τελικό αποτέλεσμα. Αν θέλουμε να εμφανιστούν όλες βάζουμε «*» αστερίσκο.
- FROM: ακολουθούν τα ονόματα των πινάκων από όπου θα διαβάσουμε τα δεδομένα.
- WHERE: ακολουθούν λογικές συνθήκες σύμφωνα με τις οποίες περιορίζουμε την εμφάνιση δεδομένων. Π.χ. παραπάνω ζητήσαμε να περιοριστούν τα αποτελέσματα σε εκείνα όπου το πεδίο City = “New York”.

Η PHP επιτρέπει σύνδεση με μια βάση δεδομένων MySql με την εντολή:

```
Mysql_connect(servername,username,password);
```

Όπου servername τοποθετούμε το όνομα του server στον οποίο θέλουμε να συνδεθούμε. Συνήθως είναι το localhost:3306. Στο username και password βάζουμε εκείνα που επιτρέπουν την σύνδεση στον MySql server. Εκτελώντας την εντολή επιστρέφεται ένα αντικείμενο που αντιπροσωπεύει την σύνδεση. Το αντικείμενο αυτό πρέπει να το καταχωρίσουμε σε μια μεταβλητή. Πχ.

```
$con = mysql_connect(“localhost:3306”,”root”,”33333”);
```

Άρα η μεταβλητή $con αντιπροσωπεύει πλέον την ανοικτή σύνδεση. Για να κλείσουμε την σύνδεση καλούμε την εντολή:

```
mysql_close($con);
```

Για να επιλέξουμε μια συγκεκριμένη βάση δεδομένων μέσα στον εξυπηρετητή χρησιμοποιούμε την εντολή:

```
Mysql_select_db(“mydatabase”, $con) ;
```

Αν η υποτιθέμενη βάση δεδομένων περιέχει τον πίνακα Customers που περιγράψαμε προηγουμένως και θέλουμε να εκτελέσουμε το ερώτημα που εμφανίζει όλους τους πελάτες που ζουν New York σε PHP θα πρέπει να αποθηκεύσουμε σε μορφή string το sql ερώτημα και μετά να καλέσουμε την mysql_query() δηλ.

```
$myQuery = “SELECT * FROM Customers WHERE City=”New York” “;

$result = mysql_query($myQuery);
```

Στην μεταβλητή $result υπάρχουν τα αποτελέσματα από την εκτέλεση του ερωτήματος. Χρησιμοποιώντας την mysql_fetch_array($result) παίρνουμε μία-μία γραμμή των αποτελεσμάτων και την επιστρέφουμε σαν array στο πρόγραμμα ώστε να μπορεί να την επεξεργαστεί.

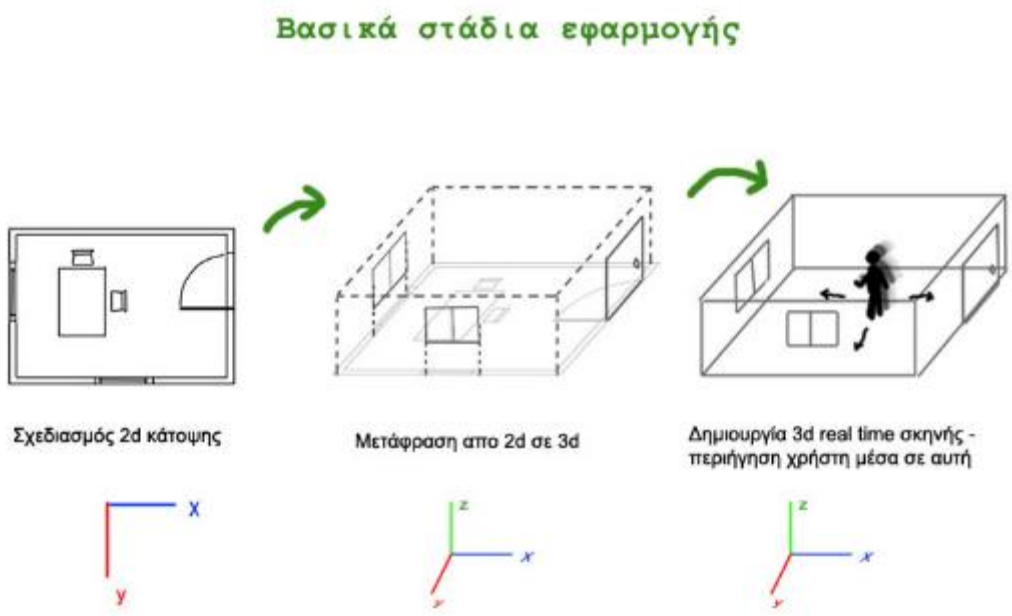
ΚΕΦΑΛΑΙΟ 4
ΣΧΕΔΙΑΣΜΟΣ ΠΡΟΓΡΑΜΜΑΤΟΣ ΚΑΙ ΓΡΑΦΙΚΗΣ ΔΙΕΠΑΦΗΣ ΧΡΗΣΤΗ

4.1 Εισαγωγή

Στο προηγούμενο κεφάλαιο περιγράψαμε τις τεχνολογίες οι οποίες χρησιμοποιούνται στην υλοποίηση των επί μέρους τμημάτων της εφαρμογής μας. Στο παρόν κεφάλαιο θα παρουσιάσουμε την φιλοσοφία και την διαδικασία σχεδιασμού της εφαρμογής από τα πρώτα θεωρητικά στάδια. Επίσης παράλληλα θα κάνουμε παρουσίαση του γραφικού περιβάλλοντος της εφαρμογής και με ποιούς τρόπους παρέχει στον χρήστη πρόσβαση στις διάφορες λειτουργίες που υποστηρίζονται από την εφαρμογή.

4.2 Φιλοσοφία και ιεράρχηση βασικών στόχων της εφαρμογής μας

Πριν ξεκινήσουμε την υλοποίηση της εφαρμογής μας προβήκαμε στην ιεράρχηση των βασικών στόχων/λειτουργιών τους οποίους θέλουμε να εκπληρώνει. Ο βασικότερος και δυσκολότερος προς υλοποίηση στόχος της εφαρμογής είναι να παρέχει ένα ολοκληρωμένο και αυτόνομο περιβάλλον σχεδίασης περιεχομένου – συγκεκριμένα σχεδίασης δισδιάστατων κατόψεων. Εξ αρχής, δόθηκε μεγάλη βάση στο να δημιουργηθούν κατάλληλα εργαλεία και interfaces, που να επιτρέπουν την δημιουργία και επεξεργασία δισδιάστατων σχεδίων από τον χρήστη. Κατόπιν, σαν δεύτερος στόχος, προκύπτει η σωστή μετάφραση των δισδιάστατων σχεδίων σε τρισδιάστατη γεωμετρία. Αυτοί οι δύο βασικοί στόχοι πλαισιώνονται από άλλους μικρότερους (την υποστήριξη εγγραφής/σύνδεσης χρηστών στο site, την λεπτομερειακή υποστήριξη βάσης δεδομένων με πληροφορίες για τα αντικείμενα διακόσμησης κτλ). Στο παρακάτω σχεδιάγραμμα συνοψίζεται με βασικό τρόπο η ταυτότητα της εφαρμογής μας:



Εικόνα 38 – περιγραφή των βασικών σχεδίων της εφαρμογής μας.

Δηλαδή στην ουσία έχουμε τρία βασικά στάδια κατά τα οποία ο χρήστης αλληλεπιδρά με την εφαρμογή (Εικόνα 38):

Σχεδίαση Δισδιάστατης Γεωμετρίας -> Μετάφραση Δισδιάστατης Γεωμετρίας σε Τρισδιάστατη -> Παρουσίαση της τρισδιάστατης σκηνής και περιήγηση σε αυτή

Οι μικρότερες λειτουργίες που πλαισιώνουν τις τρείς αυτές είναι:

- Εγγραφή του χρήστη στην ιστοσελίδα της εφαρμογής.
- Σύνδεση του χρήστη με την εφαρμογή.
- Δημιουργία Προφίλ.
- Περιήγηση στα δισδιάστατα σχέδια κατόψεων.
- Περιήγηση στις τρισδιάστατες αναπαραστάσεις σχεδίων.
- Επικοινωνία μέσω άλλων χρηστών αφήνοντας σχόλια στα σχέδιά τους ή βαθμολογώντας τα.
- Περιήγηση στις σελίδες παρουσίασης των αντικειμένων διακόσμησης.

Παρακάτω θα εξετάσουμε μία προς μία τις παραπάνω λειτουργίες και το πώς σχεδιάστηκε η υλοποίησή τους.

4.3 Δημιουργώντας γραφική διεπαφή με στοιχεία HTML

Όπως σε μια desktop εφαρμογή χρησιμοποιούνται ειδικές βιβλιοθήκες και ρουτίνες για την δημιουργία της γραφικής διεπαφής (Windows Forms σε Microsoft .Net, Swing σε Java και Microsoft Foundation Classes σε C++), έτσι και μια διαδικτυακή εφαρμογή απαιτεί ειδικές τεχνολογίες και διεργασίες για την δημιουργία γραφικών διεπαφών/GUI's.

Καθώς η διαδικτυακή εφαρμογή εκτελείται πλήρως σε περιβάλλον φυλλομετρητή, θέλοντας και μη το γραφικό περιβάλλον θα στηριχθεί σε σελίδες HTML. Ότι δυναμικές διεργασίες και αν εκτελούνται στον φόντο του φυλλομετρητή (π.χ. εκτέλεση κώδικα Javascript), ότι και αν εκτελείται στην πλευρά του εξυπηρετητή (εκτέλεση κώδικα PHP, επικοινωνία με τον server της βάσης δεδομένων) τα τελικά αποτελέσματα ενσωματώνονται σαν στοιχεία μέσα σε σελίδες HTML, οι οποίες αποτελούν την «βιτρίνα» της web application. Είναι το πρώτο στοιχείο που παρεμβάλλεται μεταξύ χρήστη και των λογικών εκτελέσιμων μονάδων της εφαρμογής. Όπως αναφέραμε στα προηγούμενα κεφάλαια κάθε HTML σελίδα αποτελείται από html elements δηλ. στοιχεία κειμένου, εικόνες, στοιχεία φορμών (κουμπιά, πλαίσια κειμένου …) κτλ. Κάθε html element μπορεί να αποτελέσει ενεργό τμήμα της γραφικής διεπαφής. Παραδείγματος χάριν μπορεί μια απλή παράγραφος <p> σε HTML να έχει συγκεκριμένο id, το οποίο χρησιμοποιείται από τον κώδικα έτσι ώστε το κείμενό της να ενημερώνεται δυναμικά από κείμενο το οποίο είναι αποθηκευμένο σε βάση δεδομένων. Το κείμενο της παραγράφου, καθώς επίσης και οι κανόνες CSS που διέπουν την εμφάνισή του, μπορούν να αλλαχθούν από κώδικα JAVASCRIPT ώστε να παρουσιάζεται ένα μήνυμα στον χρήστη σε συγκεκριμένες στιγμές. Επίσης ένα στοιχείο εικόνας σε HTML, με συγκεκριμένο id, μπορεί να χρησιμοποιείται από τον κώδικα, ώστε να καλείται συνάρτηση όταν ο χρήστης κάνει κλικ πάνω του, καθιστώντας το έτσι, ενεργό κομμάτι της γραφικής διεπαφής. Με παρόμοιο τρόπο μπορούν να προγραμματιστούν εμφανίσεις και αποκρύψεις περιοχών της σελίδας, συγκεκριμένα αντικειμένων , δίνοντας έτσι δυνατότητες δημιουργίας πλαισίων διαλόγων. Επίσης μπορεί να προγραμματιστή η διαδραστικότητα μιας οριζόντιας λίστας εικόνων σε html έτσι ώστε να αναλάβουν τον ρόλο toolbar.

4.4 Αρχική σελίδα

Όταν ο χρήστης θα πληκτρολογεί το domain name της εφαρμογής μας (στα πλαίσια αυτής της εργασίας δεν υπάρχει domain name συνδεδεμένο με τον ιστοχώρο της εφαρμογής καθώς η εκτέλεσή της γίνεται σε τοπικό επίπεδο localhost), θα μεταφέρεται στην κεντρική σελίδα της η οποία συνήθως, σε περιβάλλοντα ιστοτόπων και web εξυπηρετητών, αντιπροσωπεύεται από ένα αρχείο index.php (ή index.htm). Στην δικιά μας περίπτωση η αρχική σελίδα της εφαρμογής θα περιλαμβάνει ένα λογότυπο, μια μικρή περιγραφή των λειτουργιών της και μια μικρή φόρμα σύνδεσης του χρήστη παρέχοντας δύο πεδία Username και Password (εικόνα 39). Επίσης θα παρέχεται η δυνατότητα εγγραφής σε περίπτωση που ο χρήστης δεν είναι εγγεγραμμένος με το κουμπί σύνδεσμο : Register New User.

Εικόνα 39 – Εισαγωγική σελίδα της εφαρμογής με το πλαίσιο διαλόγου για login.

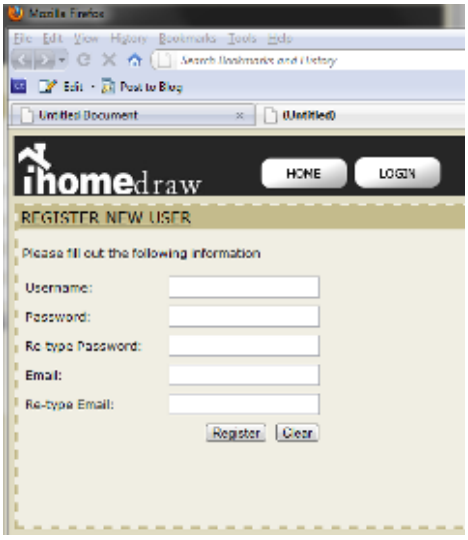
Έστω ότι ο χρήστης είναι η πρώτη φορά που επισκέπτεται τον ιστοχώρο και θέλει να εγγραφεί. Πατώντας στον σύνδεσμο Register New User θα μεταφέρεται στην σελίδα εγγραφής η οποία περιγράφεται από το αρχείο register.php.

4.5. Εγγραφή Χρήστη

Η σελίδα εγγραφής χρήστη θα απαιτεί να συμπληρώσει κάποια βασικά στοιχεία του σε μια φόρμα όπως:

- Username
- Password
- E-mail

Επίσης θα περιλαμβάνει κουμπιά Clear (για αρχικοποίηση των πεδίων της φόρμας σε περίπτωση που έχει γίνει λάθος) και Submit για υποβολή της εγγραφής (εικόνα 40).



Εικόνα 40 – Σελίδα εγγραφής χρήστη.

Καθώς ο χρήστης θα πληκτρολογεί το username του, ασύγχρονα θα γίνεται επικοινωνία με την βάση δεδομένων της εφαρμογής, για να διαπιστωθεί αν υπάρχει ήδη εγγεγραμμένος χρήστης που έχει επιλέξει το συγκεκριμένο username. Αν όντως υπάρχει, θα εμφανίζεται διακριτικό μήνυμα κοντά στο πεδίο μέχρι να πληκτρολογηθεί username το οποίο δεν υπάρχει στην βάση. Το ίδιο θα συμβεί και με το email διότι, δεν θέλουμε διπλές εγγραφές χρηστών από τον ίδιο λογαριασμό email. Επίσης, κατά την πληκτρολόγηση του password, θα γίνεται έλεγχος αν το συνθηματικό είναι αρκετά περίπλοκο ώστε να παρέχει ασφάλεια στον χρήστη. Εφόσον τηρηθούν τα κριτήρια για την σωστή εισαγωγή των βασικών στοιχείων του χρήστη, πατώντας submit θα γίνει υποβολή της αίτησης εγγραφής και ταυτόχρονα θα σταλθεί από την εφαρμογή στο email που δήλωσε ο χρήστης ένα μήνυμα με τον κωδικό ενεργοποίησης του λογαριασμού του. Ο χρήστης, πατώντας εκεί, θα μεταφέρεται σε μια ειδική σελίδα της εφαρμογής με το μήνυμα ότι ο λογαριασμός του ενεργοποιήθηκε και με σύνδεσμο που θα τον οδηγεί στην κεντρική σελίδα της εφαρμογής ώστε να κάνει login.

4.6 Login Χρήστη

Ο χρήστης όντας στην αρχική σελίδα συμπληρώνει τα στοιχεία εισόδου και πραγματοποιεί επιτυχές login. Αυτόματα θα μεταφέρεται στην σελίδα του προσωπικού του προφίλ. Εκεί θα μπορεί να έχει μια εποπτεία των στοιχείων προφίλ που έχει συμπληρώσει, προσωπικών μηνυμάτων από άλλους χρήστες, τα σχέδια που έχει αποθηκεύσει κτλ. Αν δεν έχει δημιουργήσει βασικό προφίλ θα υπάρχει μήνυμα που θα του παρέχει σύνδεσμο ώστε να προβεί στην δημιουργία νέου.

4.7 Δημιουργία προφίλ

Όταν ο χρήστης προβαίνει στην δημιουργία προσωπικού προφίλ, οδηγείται στην αντίστοιχη σελίδα η οποία περιγράφεται από το αρχείο create_profile.PHP. Η σελίδα περιλαμβάνει μια μεγάλη φόρμα με διάφορα πεδία προς συμπλήρωση (εικόνα 40). Ζητούνται προαιρετικά:

- Το όνομα του χρήστη
- Το επώνυμό του
- Το τηλέφωνό του
- Χώρα καταγωγής
- Διεύθυνση
- Επάγγελμα
- Σπουδές
- Ημερομηνία Γέννησης

Αφού συμπληρωθούν σωστά τα πεδία και γίνουν οι απαραίτητοι έλεγχοι από την Javascript για το validation τους, ο χρήστης μπορεί να πατήσει submit ώστε να καταχωρηθεί το προφίλ του. Ταυτόχρονα γίνεται επικοινωνία με την βάση δεδομένων και αποθηκεύονται εκεί τα στοιχεία του στον πίνακα user_profile.

Εικόνα 41 - Σελίδα δημιουργίας προφίλ.

Αφού γίνει η καταχώριση του προσωπικού προφίλ γίνεται επιστροφή στην προσωπική σελίδα του χρήστη.

4.8 Προσωπική Σελίδα του χρήστη

Στην προσωπική σελίδα του χρήστη η οποία περιγράφεται από το αρχείο personal_page.php παρέχονται σύνδεσμοι στις βασικές λειτουργίες της εφαρμογής.

Επίσης εμφανίζονται και διάφορα στοιχεία που αφορούν τον χρήστη π.χ. παρουσιάζονται τα στοιχεία του βασικού του προφίλ κτλ (εικόνα 42). Δίνεται η επιλογή για την δημιουργία νέου σχεδίου: Create New Design. Επιπλέον περιλαμβάνεται λίστα με τα αποθηκευμένα σχέδια του χρήστη και επιλογή Browse Designs from other Users, η οποία επιτρέπει την επισκόπηση σχεδίων από άλλους χρήστες. Κάνοντας κλικ πάνω στο όνομα ενός σχεδίου που έχει αποθηκεύσει ο χρήστης εμφανίζονται τρείς επιλογές :

- Edit – Άνοιγμα του σχεδίου για επεξεργασία
- Explore 3d – Περιήγηση στην τρισδιάστατη αναπαράσταση του σχεδίου
- Delete – Διαγραφή του σχεδίου
- Rename – Μετονομασία του σχεδίου

Επίσης υπάρχει σύνδεσμος Browse Objects η οποία μας μεταφέρει στην σελίδα επισκόπησης των αντικειμένων διακόσμησης που υπάρχουν στο πρόγραμμά μας.

Εικόνα 42 – Στιγμιότυπο από την προσωπική σελίδα του χρήστη.

4.9 Σελίδα επισκόπησης αντικειμένων διακόσμησης

Στην βάση δεδομένων της εφαρμογής, περιλαμβάνεται πίνακας με καταχωρημένες πληροφορίες για κάθε αντικείμενο διακόσμησης που υποστηρίζεται. Μέσω της συγκεκριμένης σελίδας (καθώς και μέσα από την σελίδα σχεδίασης κάτοψης) ο χρήστης μπορεί να αναζητήσει τα αντικείμενα με βάση την ονομασία, τον τύπο, με βάση την χρήση, τον κατασκευαστή κτλ. Κάνοντας κλικ σε ένα συγκεκριμένο αντικείμενο ανοίγει πλαίσιο με περισσότερες πληροφορίες.

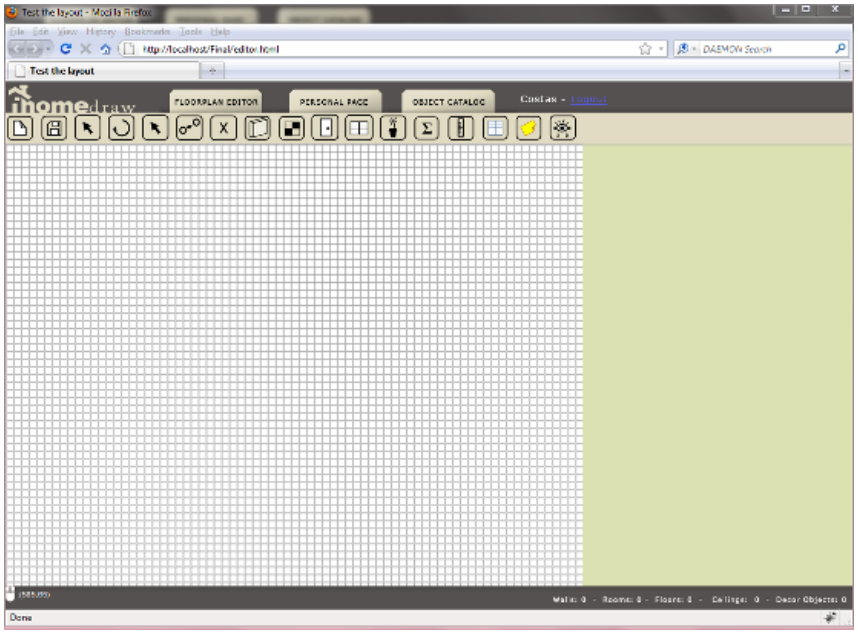
4.10 Σελίδα επισκόπησης σχεδίων από άλλους χρήστες

Στην συγκεκριμένη σελίδα ο χρήστης μπορεί να αναζητήσει σχέδια με βάση τον χρήστη ή με λέξεις κλειδιά. Για κάθε σχέδιο του δίνονται οι εξής επιλογές:

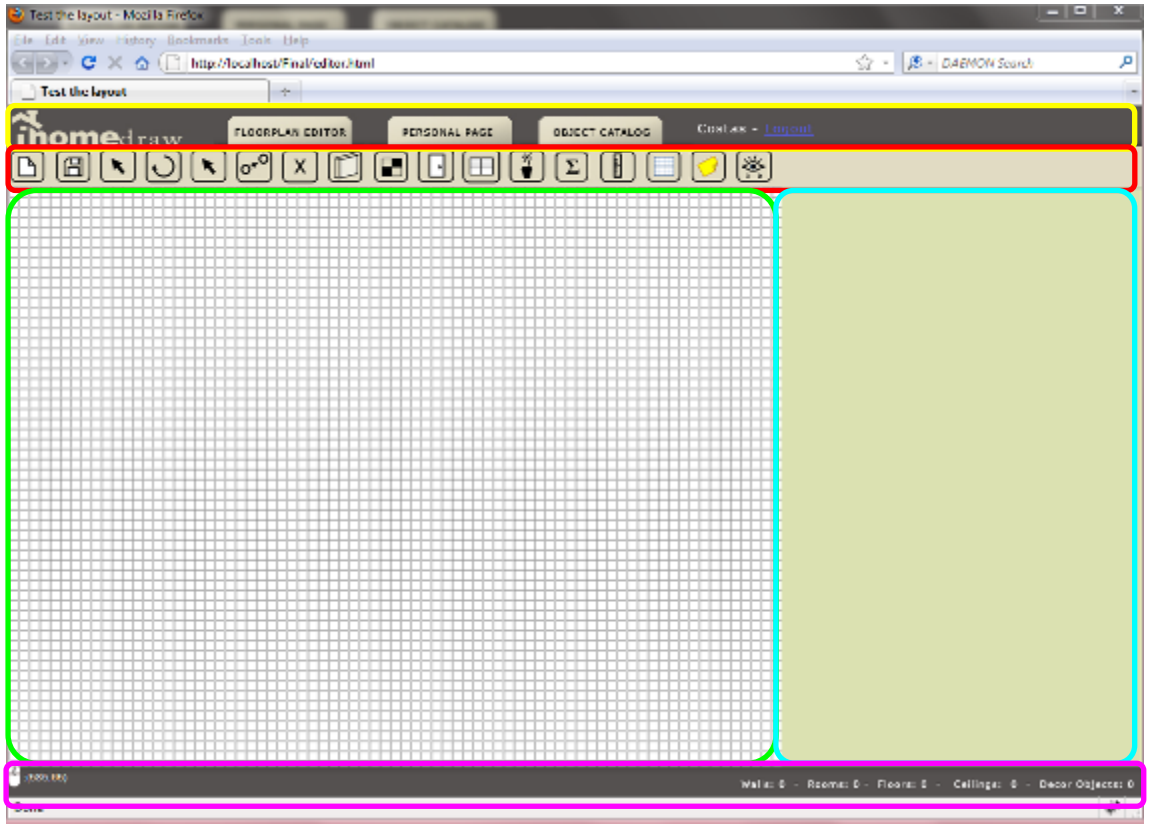
- Επισκόπηση σε 2d – Χωρίς την δυνατότητα επεξεργασίας.
- Επισκόπηση σε 3d – Περιήγηση στον τρισδιάστατο χώρο που περιγράφει το σχέδιο.
- Σχολιασμός – Τοποθέτηση σχόλιου για το συγκεκριμένο σχέδιο.
- Βαθμολογία – Απόδοση βαθμολογίας από 1 έως 5.

4.11 Δημιουργία/ Επεξεργασία δισδιάστατου σχεδίου κατόψεως

Το τμήμα αυτό της εφαρμογής θα μπορούσε να αποτελέσει ξεχωριστή εφαρμογή από μόνο του. Στην ουσία αποτελεί έναν απλοποιημένο cad editor που επιτρέπει τον σχεδιασμό δωματίων, παραθύρων, πορτών, επιπέδων και την τοποθέτηση αντικειμένων (εικόνα 43). Ολόκληρη η διεπαφή του editor υλοποιείται σε μία σελίδα την editor.php. Στην συγκεκριμένη σελίδα χρησιμοποιούμε αρκετά για να χωρίσουμε την γραφική διεπαφή σε 5 βασικές ζώνες (εικόνα 44).



Εικόνα 43- Η αρχική σελίδα του editor σχεδίασης δισδιάστατων κατόψεων.



Εικόνα 44 – Οι 5 ζώνες του interface του editor δισδιάστατων κατόψεων. Με κίτρινο σημειώνεται η ζώνη επικεφαλίδας. Με κόκκινο σημειώνεται η μπάρα εργαλείων. Με γαλάζιο σημειώνεται η πλευρική μπάρα. Με πράσινο σημειώνεται η περιοχή σχεδίασης. Με φούξια σημειώνεται η ζώνη της υποκεφαλίδας.

Στην ζώνη της επικεφαλίδας η οποία περιλαμβάνει το λογότυπο της εφαρμογής, την ονομασία του αρχείου καθώς επίσης και συνδέσμους για την μεταφορά μας στα άλλα τμήματα της εφαρμογής που ήδη περιγράψαμε.

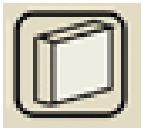
Κάτω από την ζώνη της επικεφαλίδας περιλαμβάνεται η ζώνη της μπάρας εργαλείων η οποία στην ουσία αποτελεί μια λίστα από κουμπιά τα οποία δίνουν στον χρήστη συγκεκριμένες δυνατότητες σχεδιασμού. Η μπάρα εργαλείων στην ουσία υλοποιείται από μια λίστα εικόνων html, στις οποίες έχει δοθεί διαδραστικότητα μέσω της Javascript. Κάτω από την ζώνη της μπάρας εργαλείων, περιλαμβάνεται η ζώνη με την περιοχή σχεδίασης και δεξιά της η ζώνη της πλευρικής μπάρας (sidebar). Στην ζώνη σχεδίασης θα βρίσκεται ο χώρος στον οποίον ο χρήστης θα σχεδιάζει και θα τοποθετεί αντικείμενα. Στην πλευρική μπάρα θα εμφανίζονται πληροφορίες και επιλογές ανάλογα με το αντικείμενο που έχει επιλέξει ο χρήστης ή ανάλογα με το εργαλείο σχεδίασης. Τέλος υπάρχει η ζώνη της υποκεφαλίδας όπου εκεί παρουσιάζονται πληροφορίες για τις τρέχουσες ενέργειες του χρήστη καθώς και στατιστικά για το σχέδιο κατόψεως.

4.11.1 Σχεδίαση τοίχων

Κατά την μελέτη και την σχεδίαση της εφαρμογής σε θεωρητικό επίπεδο θεωρήθηκε

βασικό να καθοριστεί το πώς θα υλοποιείται η σχεδίαση εσωτερικών χώρων. Τι εργαλεία π.χ. θα παρέχονται στον χρήστη και τι θα του ζητείται να καθορίσει πρώτα. Δόθηκε βάση πρωτίστως στον καθορισμό των τοιχίων. Αναλύοντας διάφορα σχέδια κατόψεως παρατηρήθηκε ότι το πρώτο πράγμα που διακρίνεται σε μια κάτοψη είναι οι χώροι και συγκεκριμένα τα δωμάτια. Τα εργαλεία σχεδίασης θα μπορούσαν να υλοποιηθούν, έχοντας στο επίκεντρο τον καθορισμό πρώτα τον δωματίων/χώρων και την τοποθέτησή τους στο χώρο. Θελήσαμε όμως να δώσουμε βάση στην σχεδίαση πρώτα των τοίχων καθώς αυτοί αποτελούν τον σκελετό του σχεδίου. Αν παρατηρήσουμε προσεκτικά στα σχέδια κατόψεων, οι εσωτερικοί χώροι οριοθετούνται από ένα δίκτυο τοίχων, οι οποίοι συνδέονται σαν μια μορφή γράφου. Υπάρχουν κόμβοι στους οποίους τελειώνουν και ξεκινούν τοίχοι. Υλοποιήθηκε ένα σύστημα το οποίο επιτρέπει την σχεδίαση τοίχων, διατηρώντας στην μνήμη μια δομή δεδομένων τύπου γράφου, που αποθηκεύει πληροφορίες για τους τοίχους και τα σημεία στα οποία ενώνονται. Έτσι δημιουργείται ένας οργανικός σκελετός ο οποίος δίνει τις εξής δυνατότητες:

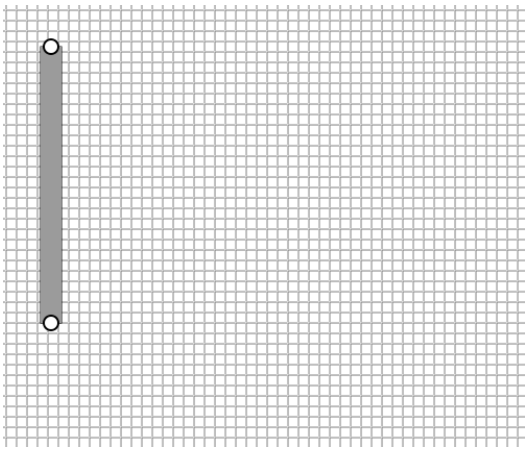
- Η μετακίνηση ενός κόμβου αλλάζει την τοποθέτηση και τον προσανατολισμό των τοίχων που συνδέονται στον κόμβο αυτό.
- Η προσθήκη ενός νέου τοίχου έχοντας σαν σημείο έναρξης έναν υπάρχον κόμβο δημιουργεί αυτόματα σύνδεση του τοίχου στον κόμβο αυτό.
- Κατά την σχεδίαση ή την μετακίνηση ενός κόμβου πάνω σε έναν ήδη υπάρχον ενώνει τα τοιχία και των δύο κόμβων σε έναν ενιαίο κόμβο.



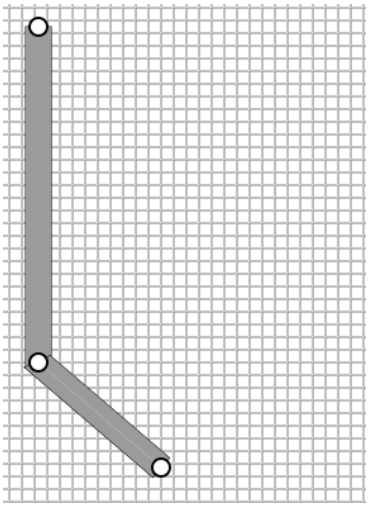
Εικόνα 45- Το εικονίδιο του εργαλείου σχεδίασης τοίχων.

Στην μπάρα εργαλείων υπάρχει το εργαλείο προσθήκης τοίχων (Εικόνα 45). Επιλέγοντάς το και έχοντας ένα άδειο σχέδιο, ο χρήστης μπορεί να κάνει κλικ οπουδήποτε στην επιφάνεια σχεδίασης και να σύρει το ποντίκι σχεδιάζοντας έναν τοίχο. Ο τοίχος αυτός έχει σαν σημείο έναρξης το σημείο στο οποίο ο χρήστης αρχικά πάτησε το κουμπί του ποντικιού και σημείο τέλους τις συντεταγμένες του τρέχοντος σημείου του κέρσορα του ποντικιού. Κατά την σχεδίαση του τοίχου εμφανίζονται και 2 μικροί κύκλοι στο σχέδιο οι οποίοι συμβολίζουν τους 2 κόμβους που ορίζουν τον τοίχο. Μετακινώντας το ποντίκι, αλλάζουν οι συντεταγμένες του δεύτερου κόμβου με μια συνεχή ανανέωση του σχήματος του τοίχου ώστε να ενώνει σε ευθεία τους δύο κόμβους. Αν αφεθεί το ποντίκι, κλειδώνει η θέση του δεύτερου κόμβου και οριστικοποιείται η τοποθέτηση του νέου τοίχου στο σχέδιο. Αν επιλεγεί ένα άλλο εργαλείο σχεδίασης, στο σχέδιο αποκρύπτονται οι «κύκλοι» που αντιπροσωπεύουν τους κόμβους και μένει μόνο το σχήμα του τοίχου. Αν επιλεγεί πάλι το εργαλείο προσθήκης τοίχου επανεμφανίζονται στο σχέδιο οι κόμβοι μεταξύ τοίχων. Αυτό γίνεται διότι μπορούν να χρησιμοποιηθούν σαν σημεία αναφοράς π.χ. κάνοντας κλικ πάνω σε έναν ήδη υπάρχοντα κύκλο επιτρέπεται η σχεδίαση ενός νέου

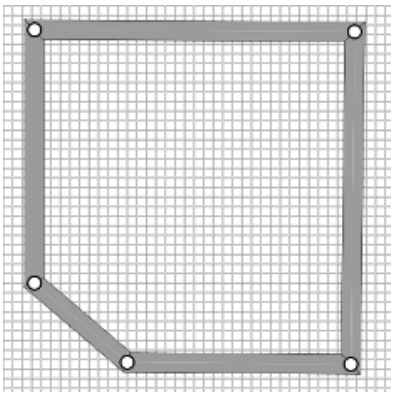
τοίχου ο οποίος ξεκινάει από τον συγκεκριμένο κόμβο και καταλήγει σε κάποιον άλλον. Επίσης κατά την σχεδίαση του τοίχου και συγκεκριμένα κατά την μετακίνηση του κόμβου που ορίζει το τέλος του, ο χρήστης μπορεί να αφήσει το ποντίκι πάνω σε έναν ήδη υπάρχων κόμβο και να συνδεθεί εκεί ο νέος τοίχος. Επίσης δίνεται η δυνατότητα στον χρήστη να ξεκινήσει την σχεδίαση ενός τοίχου από το σώμα ενός ήδη υπάρχοντος, δημιουργώντας κόμβο στο σημείο εκείνο και χωρίζοντας τον προηγούμενο τοίχο σε δύο τμήματα. Το ίδιο μπορεί να συμβεί αν κατά τον ορισμό ενός νέου τοίχου ο χρήστης σύρει και αφήσει το ποντίκι πάνω στο σώμα ενός ήδη υπάρχοντος τοίχου. Παρατηρούμε ότι δημιουργείται σε εκείνο το σημείο κόμβος ο οποίος χωρίζει τον προηγούμενο τοίχο σε δύο τμήματα αλλά και στον οποίο συνδέεται ο νέος τοίχος. Πέρα από την προσθήκη νέων τοίχων ο χρήστης πατώντας το κουμπί της επιλογής μπορεί να επιλέξει τους τοίχους που ήδη έχουν σχεδιαστεί (έναν κάθε φορά). Επιλέγοντας έναν τοίχο εμφανίζονται οι κόμβοι τους οποίους συνδέει και δίνεται η επιλογή να τους μετακινήσει αλλάζοντας το μήκος του τοίχου, τον προσανατολισμό του αλλά και επηρεάζοντας όλους τους άλλους τοίχους που συνδέονται στους δύο αυτούς κόμβους. Όταν επιλέγει έναν τοίχο στην πλευρική μπάρα εμφανίζονται λεπτομέρειες για το μήκος του, το υλικό από το οποίο είναι φτιαγμένος και το πάχος του. Δίνεται η δυνατότητα να αλλάξει το πάχος του καθώς το συγκεκριμένο πεδίο δέχεται κείμενο σε μορφή αριθμού. Επίσης δίνονται επιλογές για να καθοριστεί η υφή και το χρώμα των 2 πλευρών του τοίχου (αριστερή και δεξιά πλευρά Α και Β όπως αναφέρεται στην εφαρμογή). Παραδείγματα σχεδίασης τοίχων αναφέρονται στις εικόνες 46,47,48.



Εικόνα 46 – Έναρξη σχεδίασης τοίχου. Παρατηρούμε με άσπρους κύκλους τα δύο handles που ορίζουν την αρχή και το τέλος του τοίχου.



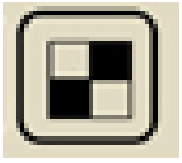
Εικόνα 47 – Προσθήκη και δεύτερου τοίχου σε σύνδεση με τον πρώτο. Παρατηρούμε ότι στον κόμβο σύνδεσης οι τοίχοι αλλάζουν σχήμα για να συνδεθούν αρμονικά στην γωνία.



Εικόνα 48 – Σχεδίαση του σκελετού ενός δωματίου. Παρατηρούμε ότι ο τελευταίος τοίχος ενώθηκε στην αρχή του πρώτου και έτσι έκλεισε το περίγραμμα του δωματίου.

4.11.2 Καθορισμός δωματίων

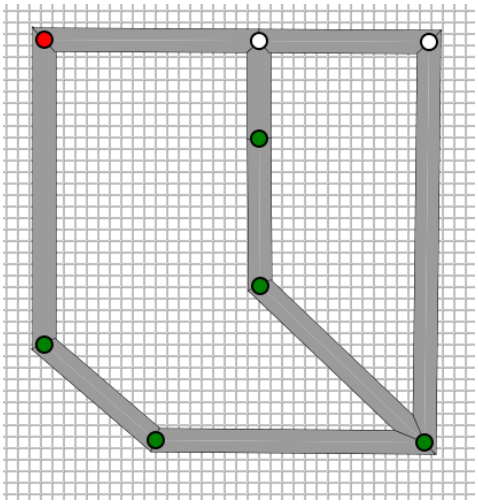
Αφού τοποθετηθούν τοίχοι στο σχέδιο μπορούν να καθοριστούν τα δωμάτια τα οποία οριοθετούν οι συγκεκριμένοι τοίχοι. Αυτό γίνεται ως εξής: Ο χρήστης επιλέγει από την μπάρα εργαλείων το εργαλείο οριοθέτησης δωματίου (εικόνα 49) και στο σχέδιο εμφανίζονται όλοι οι κόμβοι του σκελετού.



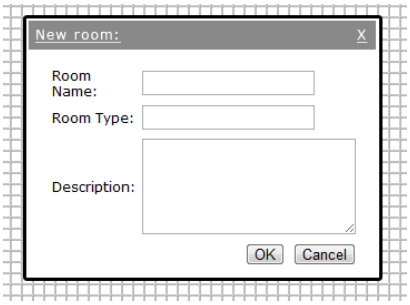
Εικόνα 49 – Το εικονίδιο του εργαλείου οριοθέτησης δωματίων.

Ο χρήστης μπορεί να επιλέξει με την σειρά τους κόμβους ανάμεσα από τους τοίχους που αποτελούν το περίγραμμα του δωματίου. Καθώς επιλέγει κόμβο με κόμβο,

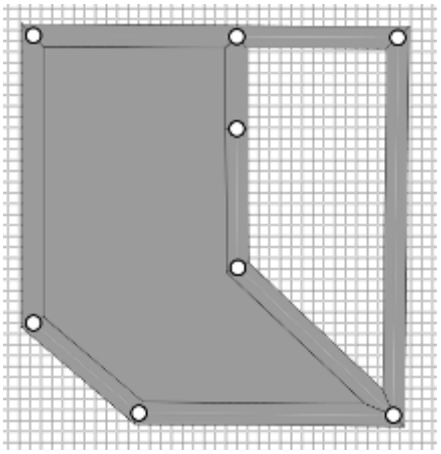
εμφανίζεται σιγά σιγά το πολύγωνο του χώρου του δωματίου μέχρι ο χρήστης να φτάσει στον αρχικό κόμβο και να κλείσει την επιλογή του (εικόνα 50,52,53). Αφού οριοθετηθεί το δωμάτιο εμφανίζεται πλαίσιο διαλόγου (εικόνα 51) που ζητάει από τον χρήστη να καθορίσει το όνομα του δωματίου και τον τύπο χρήσης του (Κουζίνα, Μπάνιο, Καθιστικό κτλ). Αφού οριοθετηθούν δωμάτια στο σχέδιο κάτοψης ο χρήστης χρησιμοποιώντας το εργαλείο επιλογής μπορεί να επιλέξει το καθένα από αυτά. Επιλέγοντας ένα δωμάτιο στην πλευρική μπάρα εμφανίζονται πληροφορίες για το μέγεθος του δωματίου, τον τύπο του, το όνομά του και δίνεται η επιλογή να επιλεχθεί υφή/χρώμα για το δάπεδο του δωματίου αλλά και για την οροφή.



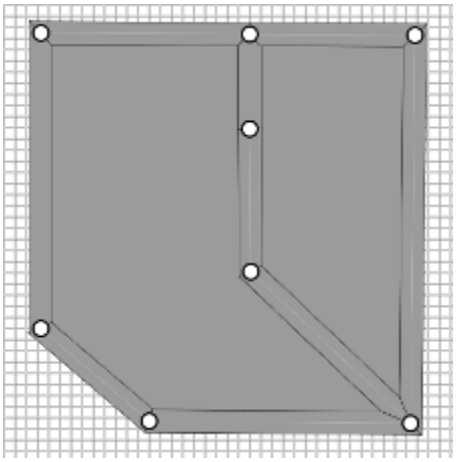
Εικόνα 50 – Αφού τοποθετήθηκαν και εσωτερικοί τοίχοι παρατηρούμε ότι το δωμάτιο χωρίστηκε στα δύο. Με το εργαλείο οριοθέτησης δωματίων ξεκινάμε επιλέγοντας κόμβο προς κόμβο να οριοθετούμε το πρώτο δωμάτιο. Παρατηρούμε ότι η έναρξη της οριοθέτησης σημειώνεται με κόκκινο στον κόμβο.



Εικόνα 51 – Πλαίσιο διαλόγου για την οριοθέτηση νέου δωματίου.



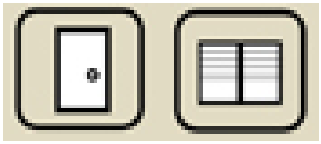
Εικόνα 52 – Καθώς κλείνει η οριοθέτηση δημιουργείται και το πάτωμα του δωματίου.



Εικόνα 53 - Ολοκληρώνοντας και την οριοθέτηση στο δεύτερο δωμάτιο.

4.11.3 Προσθήκη πορτών και παραθύρων στους τοίχους

Υπάρχουν τρία συγκεκριμένα εργαλεία στην μπάρα εργαλείων της εφαρμογής τα οποία επιτρέπουν στο χρήστη να προσθέσουν πόρτες, παράθυρα και οπές στους τοίχους του σχεδίου (εικόνα 54). Πατώντας το κουμπί προσθήκης παράθυρου εμφανίζεται στην πλευρική μπάρα μια λίστα με όλους τους διαθέσιμους τύπους παραθύρων. Ο χρήστης μπορεί να επιλέξει έναν από αυτούς.

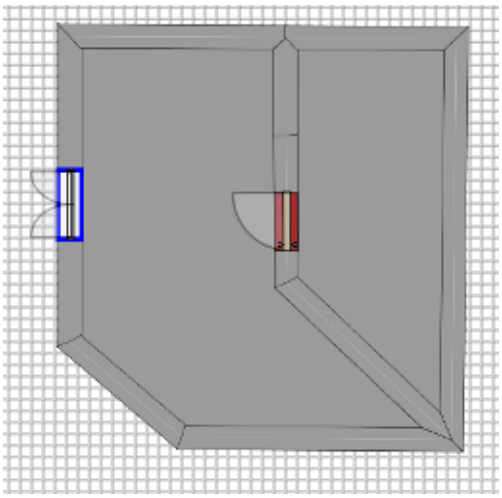


Εικόνα 54 - Τα εικονίδια των εργαλείων εισαγωγής πορτών και παραθύρων.

Πατώντας στο κουμπί insert εισάγεται στο σχέδιο το αντικείμενο του παραθύρου το οποίο χρήστης μπορεί να επιλέξει και να σύρει πάνω σε έναν τοίχο. Όταν ο χρήστης πλησιάζει το παράθυρο πάνω στον τοίχο εκείνο αναδιατάσσεται και έρχεται σε ευθεία γραμμή με την κατεύθυνση του τοίχου (align). Το ίδιο συμβαίνει χρησιμοποιώντας το

εργαλείο προσθήκης πόρτας και το εργαλείο προσθήκης οπής. Συγκεκριμένα, στο εργαλείο προσθήκης οπής, ο χρήστης μπορεί να καθορίσει το ύψος και το μήκος της τετράγωνης οπής την οποία θέλει να προσθέσει στον τοίχο. Επιλέγοντας μια πόρτα ή ένα παράθυρο στην πλευρική μπάρα εμφανίζονται όλες οι πληροφορίες σχετικά με τις διαστάσεις του αντικειμένου και το ύψος του από το έδαφος. Επίσης υπάρχει κουμπί (Side View) το οποίο ενεργοποιεί την πλευρική προβολή του τοίχου. Συγκεκριμένα εμφανίζεται ένα πλαίσιο διαλόγου με την πλάγια προβολή του τοίχου και των παραθύρων/πορτών οπών οι οποίες βρίσκονται πάνω του. Ο χρήστης μπορεί χρησιμοποιώντας το ποντίκι να επιλέξει καθένα από αυτά τα αντικείμενα και να τα μετακινήσει κατά ύψος και μήκος πάνω στον τοίχο.

Εικόνα 55 – Λίστα με της διαθέσιμες πόρτες προς εισαγωγή.



Εικόνα 56 – Τοποθετώντας πόρτες και παράθυρα.

4.11.4 Προσθήκη αντικειμένων διακόσμησης

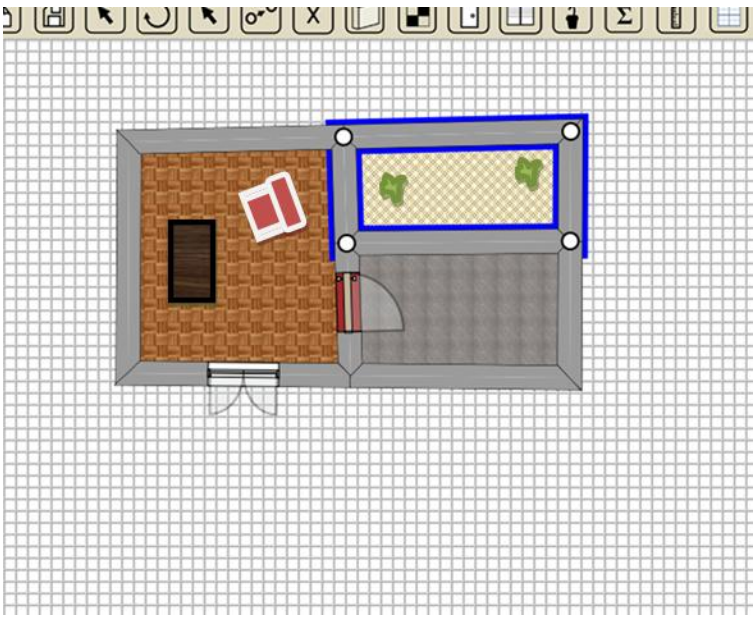
Αφού καθοριστούν τα δωμάτια, μπορούν διακοσμηθούν, τοποθετώντας μέσα τους διάφορα αντικείμενα (έπιπλα, ηλ. συσκευές, φυτά, φώτα, χαλιά κτλ). Στην μπάρα εργαλείων υπάρχει ειδικό κουμπί (εικόνα 57) για την προσθήκη αντικειμένων το οποίο εμφανίζει ένα σχετικό μενού στην πλευρική μπάρα. Το μενού αυτό περιλαμβάνει πλαίσιο αναζήτησης με βάση το όνομα του αντικειμένου και ένα

πλαίσιο επιλογής κατηγορίας αντικειμένου. Ανάλογα με τις επιλογές του χρήστη καθώς πληκτρολογεί ένα όνομα ή επιλέγει μια κατηγορία, δυναμικά η εφαρμογή επικοινωνεί με την βάση δεδομένων και ενημερώνει την λίστα εμφάνισης των αντικειμένων.

Εικόνα 57 – Το εικονίδιο του εργαλείο εισαγωγής αντικειμένων.

Επιλέγοντας ένα αντικείμενο από την λίστα, έχει ως αποτέλεσμα να εμφανιστούν παρακάτω κάποιες βασικές πληροφορίες όπως όνομα, κατασκευαστής, μια μικρή περιγραφή και κόστος. Πατώντας στο κουμπί insert, το αντικείμενο εισάγεται στο κέντρο του σχεδίου. Ο χρήστης κατόπιν μπορεί να το επιλέξει και να το μετακινήσει όπου επιθυμεί. Επιλέγοντάς το, εμφανίζονται, στην πλευρική μπάρα, πληροφορίες σχετικές με το αντικείμενο όπως και επίσης σύνδεσμος προς την σελίδα παρουσίασης του αντικειμένου (η οποία αποτελεί μια html σελίδα με περισσότερες πληροφορίες για το αντικείμενο, φωτογραφίες, βίντεο κτλ).

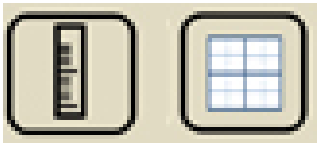
Επίσης έχοντας επιλεγμένο ένα αντικείμενο ο χρήστης μπορεί να χρησιμοποιήσει το εργαλείο περιστροφής από την μπάρα εργαλείων. Πατώντας στο αντίστοιχο εικονίδιο εμφανίζεται ο οδηγός περιστροφής γύρω από το αντικείμενο ο οποίος επιτρέπει την περιστροφή του αντικειμένου με drag ενός κυκλικού handle που εμφανίζεται γύρω από το αντικείμενο. Πληροφορείται επίσης με κείμενο για την τρέχουσα γωνία περιστροφής.



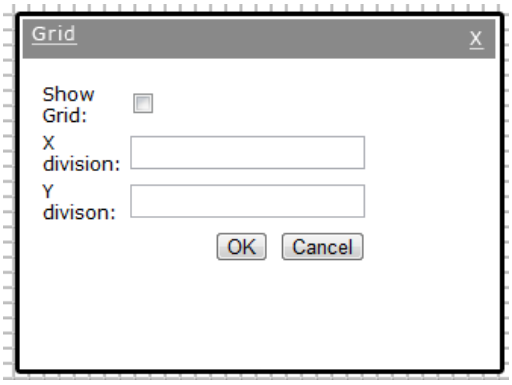
Εικόνα 58 – Τοποθέτηση επίπλων και φυτών μέσα στο σχέδιο της κάτοψης.

4.11.5 Οδηγοί

Υπάρχουν εργαλεία πάνω στην μπάρα εργαλείων για την απόκρυψη/εμφάνιση οδηγών κατά την σχεδίαση καθώς και για την αλλαγή ρυθμίσεων που αφορούν σε αυτούς (εικόνες 59,60). Καταρχήν υπάρχει εικονίδιο για την εμφάνιση απόκρυψη αποστάσεων κατά την σχεδίαση τοίχων και δωματίων. Επίσης υπάρχει εργαλείο για την εμφάνιση/ απόκρυψη μεγέθους επιφανειών δωματίων. Τέλος υπάρχει εικονίδιο για την εμφάνιση απόκρυψη πλέγματος σχεδίασης (grid). Επίσης υπάρχει εικονίδιο το οποίο εμφανίζει ένα πλαίσιο διαλόγου για να αλλαχθούν οι ρυθμίσεις σχετικά με τους οδηγούς, όπως μονάδες μέτρησης, μονάδες υποδιαίρεσης του πλέγματος κτλ.



Εικόνα 59 – Τα εικονίδια των εργαλείων που αφορούν τις ρυθμίσεις των οδηγών.

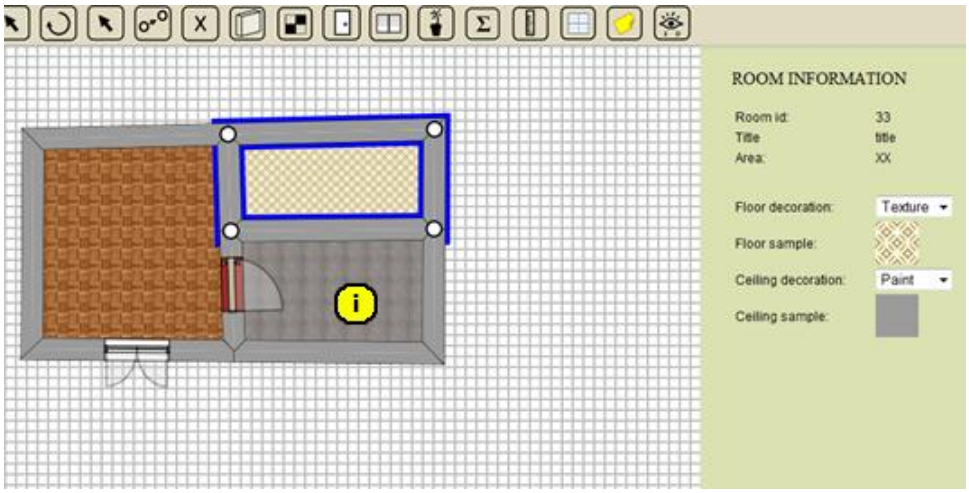


Εικόνα 60 – Επιλογές του πλέγματος

4.11.6 Εισαγωγή σημειώσεων

Πατώντας στο εικονίδιο εισαγωγής σημείωσης (εικόνα 61), ο χρήστης μπορεί να κάνει κλικ σε ένα οποιοδήποτε σημείο του σχεδίου και να προσθέσει μια σημείωση η οποία θα εμφανίζεται με το εικονιδιάκι (i) (εικόνα 62). Περνώντας το ποντίκι πάνω από το εικονίδιο θα εμφανίζεται το κείμενο της σημείωσης το οποίο μπορεί να αλλάξει από την πλευρική μπάρα. Επίσης μπορεί να μετακινήσει το εικονίδιο της σημείωσης σε άλλο σημείο πάνω στο σχέδιο.

Εικόνα 61 – Το εικονίδιο του εργαλείου εισαγωγής σημειώσεων



Εικόνα 62 – Απόσπασμα από την τοποθέτηση σημείωσης μέσα σε ένα απλό σχέδιο κάτοψης

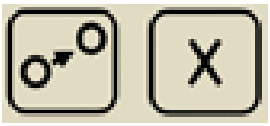
4.11.7 Εμφάνιση Κόστους και στατιστικών

Πατώντας το ειδικό πλήκτρο στην μπάρα εργαλείων της εφαρμογής (εικόνα 63) εμφανίζεται ένα ειδικό πλαίσιο διαλόγου με μια λίστα δωματίων και αντικειμένων από τα οποία αποτελείται το σχέδιο, τα επιμέρους κόστη αυτών αλλά και το συνολικό κόστος όλων.

Εικόνα 63 – Το εικονίδιο του εργαλείου προβολής κόστους και στατιστικών.

4.11.8 Αντιγραφή και διαγραφή αντικειμένων

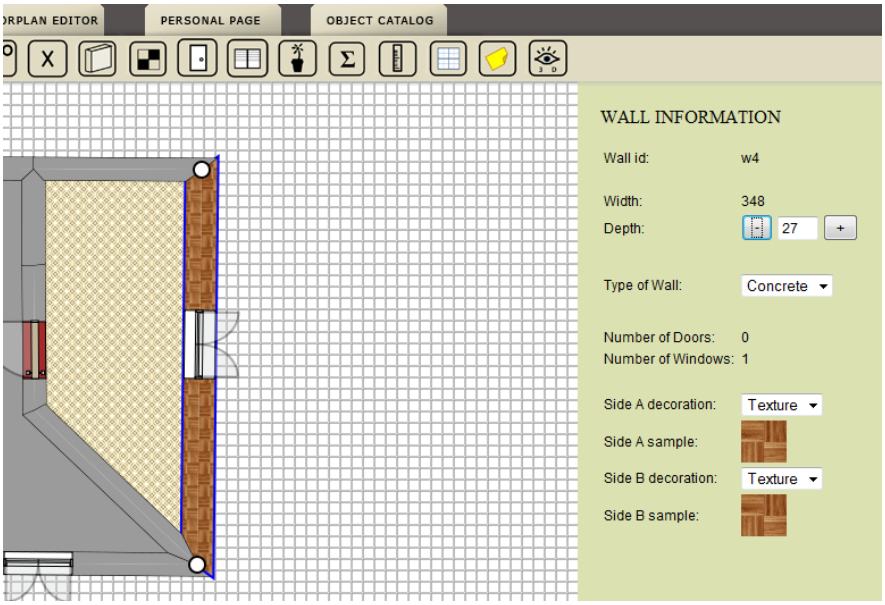
Όταν επιλέγεται ένα αντικείμενο στον χώρο σχεδίασης, υπάρχουν εργαλεία που επιτρέπουν την αντιγραφή του (δημιουργώντας λίγο πιο δίπλα, ένα αντίγραφο του αντικειμένου) ή την διαγραφή του (εικόνα 64).



Εικόνα 64 - Τα εικονίδια των εργαλείων αντιγραφής και διαγραφής

4.11.9 Επιλογή υφών

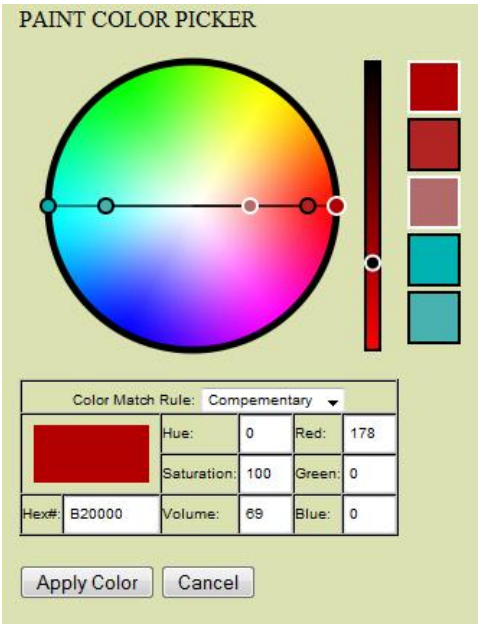
Όταν επιλέγεται ένας τοίχος ή ένα δωμάτιο δίνεται η δυνατότητα να οριστεί η διακόσμηση κάποιον επιφανειών επιλέγοντας μεταξύ υφών ή απλών χρωμάτων. Αν επιλεχθεί υφή τότε στην πλευρική μπάρα εμφανίζεται μια λίστα με τις διαθέσιμες υφές. Επιλέγοντας μια από αυτές εμφανίζονται κάποιες βασικές πληροφορίες καθώς και το κουμπί apply για εφαρμογή της υφής στην επιφάνεια του επιλεγμένου αντικειμένου (εικόνα 65).



Εικόνα 65- Πληροφορίες στην πλευρική στήλη για τον επιλεγμένο τοίχο

4.11.10 Επιλογή χρωμάτων

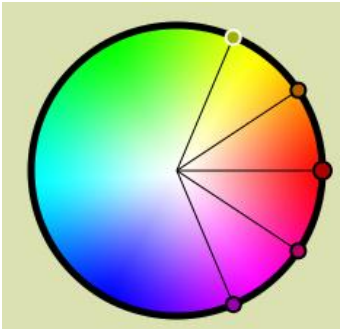
Η εφαρμογή χρωμάτων στις επιφάνειες είναι παρόμοια με την εφαρμογή υφών που περιγράψαμε στην παραπάνω παράγραφο. Όταν απαιτείται επιλογή χρώματος εμφανίζεται ένα ειδικό πλαίσιο διαλόγου στην πλευρική μπάρα το οποίο υλοποιεί έναν ισχυρό επιλογέα χρωμάτων (color picker,εικόνα 66).



Εικόνα 66 – Ο επιλογέας χρωμάτων

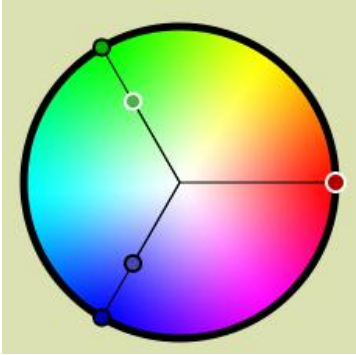
Ο επιλογέας αυτός, αποτελείται από έναν κύκλο με το φάσμα των χρωματικών αποχρώσεων. Όσο μετακινούμαστε στις μοίρες του κύκλου μεταβάλλεται η απόχρωση και όσο μετακινούμαστε πάνω στην ακτίνα δηλ. σε απόσταση από το κέντρο του κύκλου αλλάζει η ένταση της απόχρωσης. Το επιλεγμένο χρώμα εμφανίζεται σαν ένα «κυκλικό χερούλι» το οποίο μπορεί ο χρήστης να μετακινεί μέσα στον κύκλο και το οποίο συνδέεται με μια ευθεία γραμμή με το κέντρο του κύκλου. Εκτός από αυτό υπάρχουν άλλα 4 χερούλια τα οποία τοποθετούνται σε διαφορετικά σημεία του χρωματικού χώρου ανάλογα με την φόρμουλα επιλογής αρμονικών χρωμάτων.

- **Αναλογικός Κανόνας**. Όλα τα στελέχη επιλογής χρωμάτων βρίσκονται σε τέτοια θέση ώστε να χωρίζουν σε ανάλογες γωνίες τον χώρο μεταξύ του πρώτου και του τελευταίου στελέχους. Με αυτό τον κανόνα σε εφαρμογή επιλέγονται χρώματα τα οποία ταιριάζουν αρμονικά μεταξύ τους.



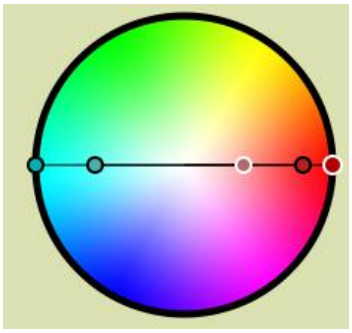
Εικόνα 67 – Αναλογικός κανόνας επιλογής αρμονικών χρωμάτων

- **Τριαδικός κανόνας.** Τα στελέχη χωρίζονται σε τρείς ομάδες οι οποίες δημιουργούν γωνίες 120 μοιρών μεταξύ τους.



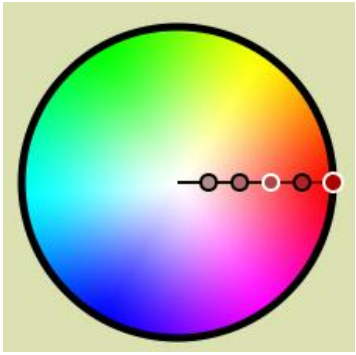
Εικόνα 68 – Τριαδικός κανόνας επιλογής αρμονικών χρωμάτων

- **Συμπληρωματικός κανόνας**. Τα στελέχη βρίσκονται πάνω στην ίδια διάμετρο δημιουργώντας συμπληρωματικές αποχρώσεις.



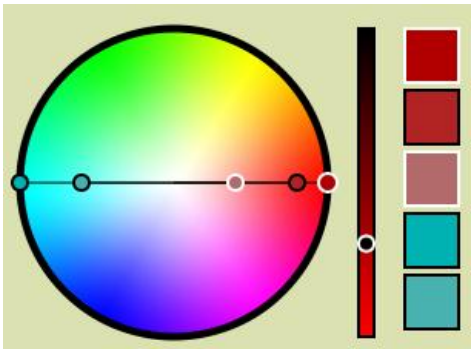
Εικόνα 69 – Συμπληρωματικός κανόνας επιλογής αρμονικών χρωμάτων

- **Μονοχρωματικός κανόνας.** Όλα τα στελέχη βρίσκονται στην ίδια ακτίνα δημιουργόντας διαβαθμίσεις της ίδιας απόχρωσης.



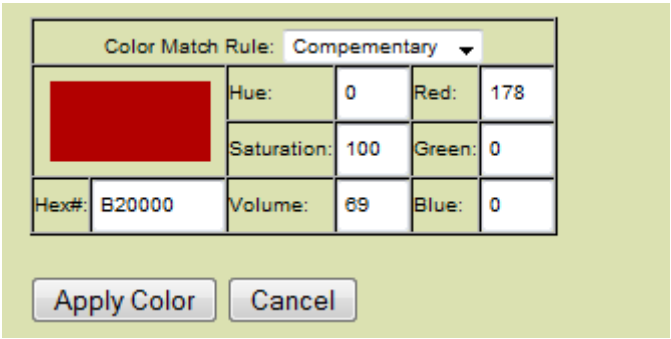
Εικόνα 70 – Μονοχρωματικός κανόνας επιλογής αρμονικών χρωμάτων

Δίπλα ακριβώς από τον κύκλο, υπάρχει μια μπάρα καθορισμού της φωτεινότητας του κάθε χρώματος και ακριβώς πιο δίπλα υπάρχουν τα 5 τετράγωνα με τα δείγματα των 5 στελεχών επιλογής χρωμάτων (εικόνα 71).



Εικόνα 71 – Η μπάρα φωτεινότητας του χρώματος και τα 5 swatches αρμονικών χρωμάτων

Πιο κάτω υπάρχει ένα πλαίσιο διαλόγου, στο οποίο ο χρήστης μπορεί να επιλέξει έναν από τους κανόνες επιλογής που αναφέρθηκαν παραπάνω. Επίσης υπάρχουν πλαίσια διαλόγου για εισαγωγή χρωματικής πληροφορίας σε σύστημα RGB , HSV αλλά και δεκαεξαδικό (εικόνα 72).



Εικόνα 72 – Αριθμητικές παράμετροι του χρώματος.

Επιλέγοντας Apply Color εφαρμόζεται το τρέχων επιλεγμένο χρώμα στην επιφάνεια. Επίσης υπάρχει και μια παλέτα στην οποία μπορούμε να αποθηκεύσουμε χρώματα που χρησιμοποιούμε συνήθως.

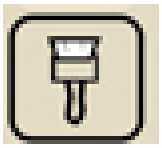
4.11.11 Εργαλείο εφαρμογής χρωμάτων και υφών

Η εφαρμογή διαθέτει εργαλείο το οποίο επιτρέπει την επιλογή μιας υφής και κατόπιν την εφαρμογή της με συνεχή κλικ στο σχέδιο σε ένα σύνολο αντικειμένων. Επιλέγοντας το εργαλείο (εικόνα 73) ο χρήστης εμφανίζει ένα πλαίσιο επιλογών στην πλευρική μπάρα στο οποίο του δίνεται η δυνατότητα για ταχεία εφαρμογή υφής/χρώματος σε επιφάνεια:

- Τοίχου
- Δαπέδου

- Οροφής

Αφού επιλέξει κατηγορία εφαρμογής από κάτω επιλέγει χρώμα ή υφή και εμφανίζεται το τρέχων δείγμα. Κατόπιν μεταφέροντας τον κέρσορα στο σχέδιο καθώς περνάει πάνω από αντικείμενα στα οποία μπορεί να εφαρμοστεί η υφή/χρώμα εκείνα μαρκάρονται και κάνοντας κλικ πάνω τους εφαρμόζεται η αλλαγή.



Εικόνα 73 – Εικονίδιο εργαλείου εφαρμογής χρωμάτων και υφών.

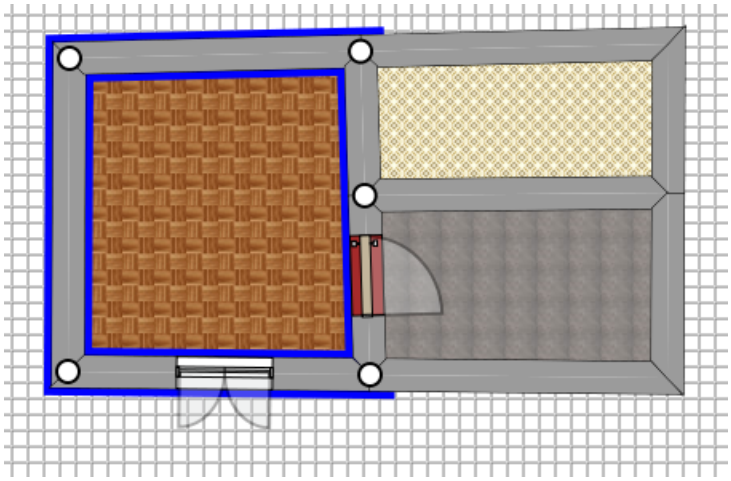
4.11.12 Αποθηκεύοντας το σχέδιο

Πατώντας στο εικονίδιο με την δισκέτα δίνεται η επιλογή στον χρήστη να αποθηκεύσει το σχέδιο στον διακομιστή και να του δώσει όνομα καθώς και μερικές λέξεις κλειδιά για την αναζήτηση του σχεδίου από άλλους χρήστες. Πατώντας στο εικονίδιο με τον φάκελο ο χρήστης μπορεί να ανοίξει ένα άλλο σχέδιο που έχει ήδη δημιουργήσει για επεξεργασία.

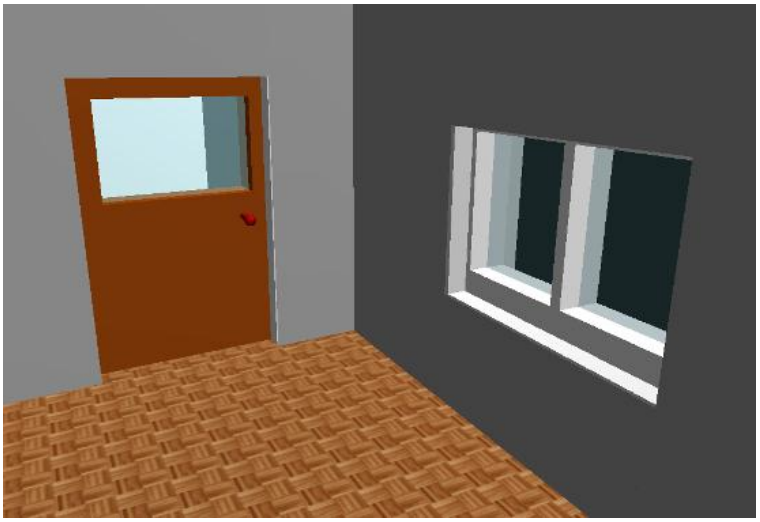
4.11.13 Παράμετροι Σχεδίου

Πατώντας στο ειδικό εικονίδιο εμφανίζεται στην πλευρική μπάρα ένα πλαίσιο διαλόγου για την ρύθμιση κάποιων βασικών παραμέτρων του σχεδίου. Δίνεται η δυνατότητα να προστεθεί ένας τίτλος και μια περιγραφή καθώς και να καθοριστεί σε πιο σημείο του σχεδίου θα ξεκινάει η περιήγηση του χρήστη όταν θα δημιουργηθεί η τρισδιάστατη σκηνή.

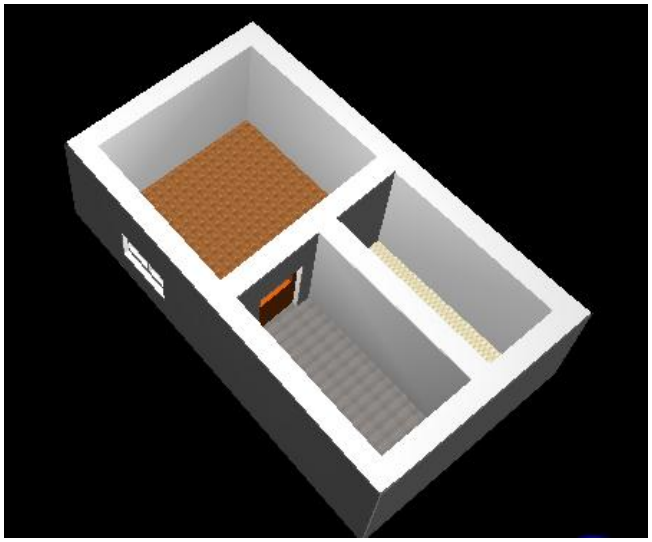
4.12 Δημιουργία τρισδιάστατης σκηνής και περιήγηση μέσα σε αυτή



Εικόνα 74 – Ένα απλό σχέδιο κάτοψης



Εικόνα 75 – Σκηνή από την περιήγηση στον τρισδιάστατο χώρο του σχεδίου της εικόνας 36



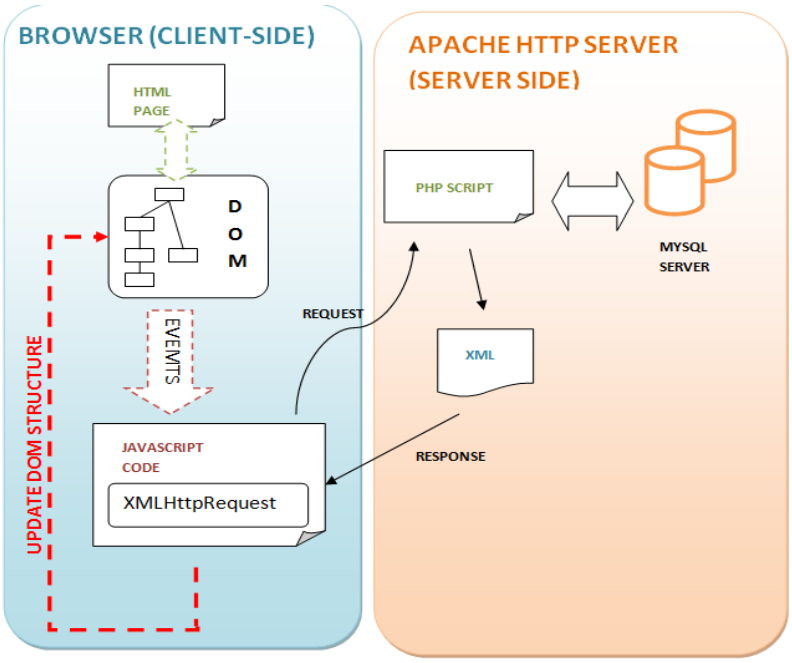
Εικόνα 76 – Το τρισδιάστατο μοντέλο του απλού σχεδίου της εικόνας 36. Παρατηρούμε την αντιστοίχηση των υφών στα δάπεδα.

Κατά την αποθήκευση του σχεδίου δημιουργείται αυτόματα το αρχείο με την τρισδιάστατη γεωμετρία της σκηνής που περιγράφει η κάτοψη. Η περιήγηση γίνεται μέσω της ειδικής σελίδας Explore in 3d η οποία περιγράφεται από το αρχείο explore.PHP. Στην σελίδα αυτή μέσω plug-in φορτώνεται το αρχείο X3D που περιγράφει την σκηνή και επιτρέπεται η περιήγηση μέσα σε αυτήν μέσω controls που προσφέρει το X3D plugin (εικόνες 74,75,76).

ΚΕΦΑΛΑΙΟ 5
ΥΛΟΠΟΙΗΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

5.1 Εισαγωγή

Στο κεφάλαιο αυτό θα περιγραφούν αναλυτικά οι μεθοδολογίες, τα στάδια, οι τεχνολογίες που χρησιμοποιήθηκαν καθώς και η μεταξύ τους συνεργασία ώστε να επιτευχθεί η υλοποίηση της εφαρμογής. Όπως αναφέρθηκε στα προηγούμενα κεφάλαια, διαδικτυακή, είναι μια εφαρμογή η οποία εκτελείται σε περιβάλλον φυλλομετρητή, ο πυρήνας της όμως βρίσκεται σε έναν διακομιστή ιστού (web server). Ο εκτελέσιμος κώδικας της εφαρμογής χωρίζεται, στο κομμάτι του κώδικα που εκτελείται στην πλευρά του φυλλομετρητή (μέσω Javascript) και στο κομμάτι κώδικα που εκτελείται στην πλευρά του διακομιστή (web-server μέσω PHP scripts). Η παρουσίαση γίνεται μέσω σελίδων html. Στην συγκεκριμένη εφαρμογή, τμήματα των σελίδων αυτών, περιέχουν στοιχεία SVG για την αναπαράσταση των δισδιάστατων γραφικών και στοιχεία X3D για την αναπαράσταση των τρισδιάστατων γραφικών. Ο κώδικας που εκτελείται στην πλευρά του εξυπηρετητή, επικοινωνεί με τον διακομιστή της βάσης δεδομένων που στην περίπτωσή μας είναι τεχνολογίας MySql. Η επικοινωνία μεταξύ κώδικα Javascript, στην πλευρά του φυλλομετρητή (περιβάλλον client) και στην πλευρά του web-server γίνεται χρησιμοποιώντας τεχνολογία Asynchronous Javascript and XML (Ajax).



Εικόνα 77 - Γραφική αναπαράσταση των δύο βασικών τμημάτων εκτέλεσης της εφαρμογής (client/server).

Πιο συγκεκριμένα χρησιμοποιείται το αντικείμενο XMLHttpRequest το οποίο επιτρέπει την ασύγχρονη ανταλλαγή δεδομένων με τον διακομιστή χωρίς να γίνεται ανανέωση της σελίδας. Η ανταλλαγή των δεδομένων γίνεται με την μορφή εγγράφων, μορφοποιημένων σε XML. Στην εικόνα 78, φαίνεται ξεκάθαρα η βασική δομή λειτουργίας της εφαρμογής, τόσο σε περιβάλλον φυλλομετρητή όσο και σε περιβάλλον εξυπηρετητή. Παρατηρούμε ότι το πρώτο στάδιο επαφής με τον χρήστη είναι η σελίδα html, η οποία όμως διατηρεί ένα ενεργό Document Object Model. Ο χρήστης αλληλεπιδρά με τα στοιχεία της σελίδας τα οποία δημιουργούν events. Τα events με την σειρά τους καλούν συναρτήσεις του κώδικα Javascript. Όποτε είναι αναγκαίο (π.χ. στην φόρτωση αντικειμένων διακόσμησης), χρησιμοποιείται το XMLHttpRequest object για να στείλει ασύγχρονα ένα request στο server με κατάλληλες μεταβλητές, ζητώντας την εκτέλεση συγκεκριμένου PHP script. Το script αυτό με την σειρά του θα επικοινωνήσει με τον sql server, θα διαβάσει τα δεδομένα από τους πίνακες της βάσης, εκτελώντας sql queries και θα δομήσει τα αποτελέσματα σε μορφή XML. Αυτό το κομμάτι XML επιστρέφεται σαν response στον κώδικα Javascript, ο οποίος το αποδομεί (διασχίζοντας την δενδρική δομή του XML DOM) και καλεί κατάλληλες μεθόδους για να αλλάξει τις ιδιότητες των αντικειμένων του HTML DOM. Οι αλλαγές αυτές έχουν αντίκτυπο στην αλλαγή του περιεχομένου της σελίδας HTML στο παράθυρο του φυλλομετρητή.

5.2 Σελίδες HTML και HTML DOM

Οι σελίδες HTML παίζουν τον ρόλο γραφικής διεπαφής για τον χρήστη. Ο φυλλομετρητής διαβάζει τον κώδικα html και αναπαριστά γραφικά τα html elements στην οθόνη του υπολογιστή, εμφανίζοντας την σελίδα. Ταυτόχρονα όμως, ενώ ο χρήστης έχει φορτωμένη την σελίδα στην οθόνη του, ο φυλλομετρητής διαβάζοντας τον κώδικα html, δημιουργεί μια δενδρική δομή γνωστή ως Document Object Model. Στους κόμβους της δομής είναι αποθηκευμένα σαν αντικείμενα όλα τα στοιχεία html, με όλες τις παραμέτρους τους προσβάσιμες από τον κώδικα Javascript. Έτσι αλλάζοντας τις παραμέτρους αυτές αλλά και έχοντας δυνατότητα διαγραφής ή προσθήκης νέων κόμβων στο DOM, είναι εφικτό, δυναμικά να αλλάξει η εμφάνιση και το περιεχόμενο της σελίδας. Για να είναι δυνατή η πρόσβαση από την Javascript σε ένα στοιχείο της σελίδας, αυτό θα πρέπει να διαθέτει μια ιδιότητα id με μια ξεχωριστή ονομαστική τιμή. Παραδείγματος χάριν στην αρχική σελίδα της εφαρμογής, υπάρχουν δύο πεδία κειμένου στα οποία ο χρήστης πληκτρολογεί το username και το password για να κάνει login. Για να επιτραπεί (μέσω Javascript) άμεση πρόσβαση και στα δύο πεδία input θα πρέπει να διαθέτουν δύο χαρακτηριστικά id’s. Στο πρώτο πεδίο δίνεται το id =”txtUsername” και στο δεύτερο “txtPassword”. Επίσης στο κουμπί login δίνεται id= “btnLogin” και τοποθετείται ένας eventlistener σε περίπτωση που ο χρήστης κάνει κλικ (δηλ. onClick event). Και τα τρία στοιχεία html είναι άμεσα προσβάσιμα από την Javascript χρησιμοποιώντας την εντολή document.getElementById(“…”). Σαν όρισμα στην παρένθεση τοποθετείται το id του στοιχείου προς επεξεργασία. Το document είναι ένα αντικείμενο/object στην

γλώσσα Javascript το οποίο παρέχεται από τον φυλλομετρητή, και στην ουσία αντιπροσωπεύει την σελίδα html η οποία είναι φορτωμένη εκείνη την στιγμή. Η getElementById() επιστρέφει σε μορφή object το συγκεκριμένο html element άρα πρέπει να καταχωρηθεί σε μια μεταβλητή ώστε να είναι δυνατή η πρόσβαση στις ιδιότητες του. Π.χ.

```
Var username = document.getElementById("txtUsername");
If (username.value == ….
```

Επίσης η προσθήκη event listener γίνεται στο tag του html element χρησιμοποιώντας ένα από τα κατάλληλα event-attributes π.χ. `<input onclick="…όνομα συνάρτησης Javascript προς εκτέλεση…" >`

Στην αρχική σελίδα index.PHP χρησιμοποιείται το script login.js στο οποίο υπάρχει κώδικας που εφόσον ο χρήστης πατήσει το κουμπί login εκτελείται η μέθοδος checkLogin(). Η μέθοδος διαβάζει της τιμές των πεδίων txtUsername και txtPassword και τις στέλνει σαν μεταβλητές (μέσω XMLHttpRequest) στον server και συγκεκριμένα στο script checkLogin.PHP. Το script αυτό συνδέεται με τον MySql server και ελέγχει στον πίνακα users αν υπάρχει εγγραφή με ίδιο username και password με αυτά που πληκτρολόγησε ο χρήστης. Αν υπάρχει τότε στέλνει τα κατάλληλα δεδομένα σχετικά με τα στοιχεία του χρήστη σε μορφή xml και φορτώνει την προσωπική του σελίδα. Το ίδιο συμβαίνει και όταν π.χ. ο χρήστης θέλει να κάνει εγγραφή. Η σελίδα register.PHP περιλαμβάνει μια φόρμα εγγραφής στην οποία κάθε html element έχει ένα ξεχωριστό id. Ο κώδικας Javascript δημιουργεί αναφορές στα στοιχεία αυτά καλώντας συνεχώς την document.getElementById() και καταχωρώντας το object που επιστρέφεται σε μια μεταβλητή. Κατόπιν ενώ ο χρήστης π.χ. συμπληρώνει τα πεδία υπάρχουν events τύπου onChange τα οποία ενεργοποιούνται εφόσον αλλάξει το κείμενο στα πεδία εισόδου (π.χ. ενώ ο χρήστης πληκτρολογεί). Τα onchange events ενεργοποιούν διαδικασίες στην Javascript οι οποίες χρησιμοποιώντας το XMLHttpRequest Object επικοινωνούν ασύγχρονα με τον server, κάνοντας ελέγχους στον πίνακα users για το αν υπάρχουν ταυτίσεις με τα νέα στοιχεία που πρόκειται να εισαχθούν στην βάση. Εφόσον όλα πάνε καλά εκτελείται, πάλι μέσω XMLHttpRequest script της PHP, το οποίο δέχεται τις τιμές που έχει εισάγει ο χρήστης και κάνει INSERT στην βάση δεδομένων. Οι ίδιες τεχνικές χρησιμοποιούνται και όταν ο χρήστης επιθυμεί να δημιουργήσει προσωπικό προφίλ, ή να αφήσει σχόλια στα σχέδια άλλου χρήστη κτλ.

5.3 Υλοποίηση της λογικής του τμήματος σχεδίασης δισδιάστατων γραφικών

Ένα από τα πιο βασικά τμήματα της εφαρμογής είναι ο ενσωματωμένος editor ο οποίος επιτρέπει τον σχεδιασμό δισδιάστατων γραφικών που αντιπροσωπεύουν σχέδια κατόψεων. Η υλοποίηση αυτού του τμήματος της εφαρμογής, έγινε

χρησιμοποιώντας τεχνολογία SVG σε συνδυασμό με σελίδες HTML, Javascript και επικοινωνία με τον server μέσω τεχνολογίας Ajax. Καταρχήν η SVG, αποτελεί μόνο γλώσσα επισήμανσης το οποίο σημαίνει ότι μπορούμε μόνο να χρησιμοποιηθεί για την περιγραφή και αναπαράσταση δισδιάστατων γραφικών στην οθόνη. Θα πρέπει να δημιουργηθούν κατάλληλες δομές δεδομένων, οι οποίες να αντιπροσωπεύουν κάθε γραφικό στοιχείο ως αντικείμενο. Το αντικείμενο αυτό θα έχει συγκεκριμένες ιδιότητες, μεθόδους και λειτουργικότητα τα οποία ορίζονται από την κλάση που το περιγράφει.

5.3.1 Δημιουργία της περιοχής σχεδίασης

Στην σελίδα editor.html υπάρχει ένα στοιχείο , το οποίο έχει τοποθετηθεί για να περικλείσει την περιοχή σχεδίασης, στην οποία ο χρήστης θα μπορεί να σχεδιάζει κατόψεις. Το συγκεκριμένο έχει id=”divMain”. Το script scriptSVG.js αρχικοποιεί το περιβάλλον χρησιμοποίησης γραφικών SVG στην ιστοσελίδα.

Παρακάτω παραθέτουμε ένα κομμάτι κώδικα από το αρχείο scriptSVG.js:

ΚΩΔΙΚΑΣ #1:

```
1.  //Script to initialize SVG area

2.  // some globals

3.  var xmlns = "http://www.w3.org/2000/SVG";
4.  var xlink = "http://www.w3.org/1999/xlink";


5.  var SVGroot = null;
6.  var BackDrop = null;
7.  var SVGdefs = null;

8.  // initialize SVG
9.  function InitSVG()
10. {


11. SVGroot = document.createElementNS(xmlns,"SVG",null);

12. var A = {
13. "width":"100%",
14. "height":"100%"

15. }

16. assign(SVGroot,A);

17. document.getElementById("divMain").appendChild(SVGroot);


18. //ZoomPan = document.createElementNS(xmlns,"g",null);

19. //SVGroot.appendChild(ZoomPan);


20. BackDrop = document.createElementNS(xmlns,"rect");

21. A = {
22. "type":"backdrop",
23. "width":"100%",
24. "height":"100%",
```

```
25. "fill":"pink",
26. "pointer-events":"all"

27. }

28. assign(BackDrop,A);

29. Defs = document.createElementNS(xmlns,"defs");

30. SVGroot.appendChild(Defs);

31. var BackGroup = document.createElementNS(xmlns,"g",null)

32. BackGroup.appendChild(BackDrop);

33. SVGroot.appendChild(BackGroup);


34. TrueCoords = SVGroot.createSVGPoint();
35. GrabPoint = SVGroot.createSVGPoint();
36. OriginPoint = SVGroot.createSVGPoint();

37. }

38. // Function to assign sets of properties to objects
39. function assign(O, A)
40. {
41. for (i in A)
42. {
43. O.setAttributeNS(null,i,A[i]);
44. }
45. }
```

Πολλές φορές μέσα στις διαδικασίες της εφαρμογής δημιουργούνται νέα στοιχεία/elements (είτε HTML elements, είτε SVG elements, είτε X3D elements) στα οποία θα πρέπει να καθοριστεί ένας αριθμός από ιδιότητες/attributes. Στην Javascript για την δημιουργία ενός νέου element μέσα στην ιστοσελίδα χρησιμοποιείται η συνάρτηση document.createElementNS(namespace, element_name).

Το όρισμα namespace αναφέρεται σε ένα συγκεκριμένο namespace που καθορίζει τον τύπο του element καθώς επίσης και τι είδους Document Object Model πρέπει να δημιουργηθεί για να το περιγράψει. Παραδείγματος χάριν αν θέλουμε να προσθέσουμε ένα element τύπου SVG μέσα σε μια σελίδα HTML θα πρέπει να δημιουργηθεί και ένα SVG DOM που θα έχει αναφορά στο στοιχείο, πέρα από το HTML DOM που ήδη συντηρεί ο φυλλομετρητής για την ιστοσελίδα. Τότε κατά την κλήση της συνάρτησης createElementNS θα πρέπει να διοχετευθεί το κατάλληλο namespace, το οποίο στην περίπτωση ενός SVG στοιχείου είναι το http://www.w3.org/2000/SVG. Για να μην χρειάζεται κάθε φορά να δηλώνεται αναλυτικά το SVG namespace, αποθηκεύεται σαν αλφαριθμητικό σε μια global μεταβλητή τύπου string, η οποία είναι η εξής (γραμμή 3 του κώδικα):

```
var xmlns = "http://www.w3.org/2000/SVG";
```

Επίσης επειδή στα διανυσματικά γραφικά SVG, λόγω της άμεσης καταγωγής από την γλώσσα XML, χρησιμοποιούνται αναφορές και σύνδεσμοι που στηρίζονται στην γλώσσα XLINK, πρέπει να υπάρχει αναφορά και στο namespace της γλώσσας

XLINK, το οποίο καταχωρείται ως string σε μια global μεταβλητή (κώδικας #1, γραμμή #4):

```
var xlink = "http://www.w3.org/1999/xlink";
```

Καλώντας την document.createElementNS() δημιουργείται ένα νέο αντικείμενο το οποίο παρέχει αναφορά στο νέο στοιχείο που προσθέσαμε στην σελίδα. Το νέο αυτό αντικείμενο περιέχει attributes τις οποίες για να αλλάξουν θα να γίνεται κλήση, για κάθε μια ξεχωριστά, της μέθοδου setAttributeNS(namespace,attribute_name,attribute_value). Για να γλιτώσουμε από της επαναλαμβανόμενες κλήσεις της setAttributeNS δημιουργήσαμε μια βοηθητική συνάρτηση, την assign. Η assign δέχεται σαν ορίσματα ένα αντικείμενο και έναν πίνακα/array. Το array αυτό περιλαμβάνει για κλειδιά του πίνακα, αλφαριθμητικά τύπου "attribute name" τα οποία αντιστοιχούν στις τιμές του πίνακα που είναι αλφαριθμητικά, τύπου "attribute value". Έτσι η συνάρτηση assign (κώδικας #1, γραμμή #39) δέχεται δύο ορίσματα O και A (σημειώνουμε ότι οι μεταβλητές στην Javascript δεν χρειάζονται δήλωση τύπων γι' αυτό και στην δήλωση των ορισμάτων της συνάρτησης, αναφέρονται μόνο τα ονόματα των ορισμάτων και όχι οι τύποι όπως γίνεται π.χ. σε άλλες γλώσσες βλ. java, c). Το όρισμα O αντιπροσωπεύει το αντικείμενο στο οποίο θέλουμε να αλλάξουμε τα attributes και το A το Array που περιγράψαμε προ ολίγου. Ακολουθεί ένας βρόγχος (κώδικας #1, γραμμή #41) ο οποίος επαναλαμβάνεται για κάθε καταχώρηση στον πίνακα A. Μέσα στον βρόγχο έχουμε για κάθε κλειδί και τιμή την κλήση της συνάρτησης setAttributeNS() στο αντικείμενο O.

```
O.setAttributeNS(null,i,A[i]);
```

Στο όρισμα namespace, δεν χρειάζεται να δηλωθεί το namespace του στοιχείου O, καθώς ο browser το γνωρίζει ήδη από την δημιουργία του. Άρα απλώς δηλώνεται ως null. Σαν όρισμα attribute_name θέτεται το i το οποίο λόγο του βρόχου αναφέρεται στα κλειδιά του πίνακα A τα οποία είναι ονόματα attributes. Για όρισμα attribute_value τοποθετείται κατευθείαν η τιμή του συγκεκριμένου στοιχείου του πίνακα η οποία είναι ένα string που περιγράφει μια συγκεκριμένη value η οποία πρέπει να τοποθετηθεί στο συγκεκριμένο attribute. Η συνάρτηση initSVG() έχει σαν σκοπό να αρχικοποιήσει κάποια στοιχεία γραφικών SVG και να τα ενσωματώσει στην σελίδα HTML. Καταρχήν θα πρέπει να δημιουργήσει το βασικό root στοιχείο κάθε εγγράφου SVG, το οποίο είναι το SVG Element. Παρατηρούμε λοιπόν (κώδικας #1 - γραμμή #11):

```
SVGroot = document.createElementNS(xmlns,"SVG",null);
```

Γίνεται κλήση της συνάρτησης createElementNS και διοχετεύεται σαν όρισμα namespace η global μεταβλητή xmlns, που περιέχει σε string το namespace για SVG στοιχεία. Σαν element name διοχετεύεται φυσικά το "SVG", καθώς είναι γνωστό ότι στα έγγραφα SVG το root στοιχείο SVG element περιγράφεται από το tag <SVG>. Η κλήση της συγκεκριμένης συνάρτησης δημιουργεί ένα νέο object, το οποίο παρέχει αναφορά στο SVG element και το καταχωρούμε στην μεταβλητή SVGroot, η οποία είναι global. Όποτε λοιπόν χρησιμοποιείται η μεταβλητή SVGroot, γίνεται αναφορά στο στοιχείο SVG element, το οποίο δημιουργήθηκε και πρόκειται να παρέχει τον καμβά για το σύστημα σχεδίασης των δισδιάστατων γραφικών της εφαρμογής.

Επίσης (κώδικας #1, γραμμή #12) δηλώνεται ένας πίνακας A ο οποίος περιλαμβάνει τον συνδυασμό κλειδιών -> τιμών:

```
A[«width”] -> “100%”
A[“height”] -> “100%”
```

Οι παραπάνω συνδυασμοί αναφέρονται στις ιδιότητες του στοιχείου SVG Element, width και height που καθορίζουν το μήκος και το ύψος της επιφάνειας του καμβά. Θα πρέπει να καλύπτει το 100% της περιοχής που του δίνεται οπότε δηλώνεται 100% και στα δύο. Η αυτόματη αλλαγή των attributes στο στοιχείο SVGroot γίνεται καλώντας την συνάρτηση assign με ορίσματα το SVGroot και τον πίνακα A (κώδικας #1, γραμμή #16) :

```
assign(SVGroot,A);
```

Κατόπιν θα πρέπει να τοποθετηθεί το στοιχείο SVG μέσα στο σώμα της HTML σελίδας και συγκεκριμένα μέσα στο . Αυτό πραγματοποιείται με την εντολή:

```
document.getElementById("divMain").appendChild(SVGroot);
```

Καταρχήν καλώντας την document.getElementById(“divMain”) η Javascript επιστρέφει το object που αντιπροσωπεύει το στοιχείο με id = divMain. Ταυτόχρονα μπορούμε να χρησιμοποιήσουμε το «.» ώστε να έχουμε πρόσβαση στις dom μεθόδους του αντικειμένου και συγκεκριμένα στην appendChild, η οποία παίρνει σαν όρισμα το SVGroot. Αυτό σημαίνει ότι στο DOM της σελίδας HTML, το στοιχείο divMain περιέχει πλέον το στοιχείο SVGelement, το οποίο αυτόματα απεικονίζεται γραφικά στην σελίδα. Στις επόμενες γραμμές του κώδικα #1 καλούνται συναρτήσεις που έχουν ήδη περιγραφεί, για να δημιουργηθούν περαιτέρω SVG στοιχεία μέσα στον καμβά σχεδίασης SVGroot. Συγκεκριμένα δημιουργείται ένα ορθογώνιο/rectangle και αποθηκεύεται αναφορά στην global μεταβλητή Backdrop. Το συγκεκριμένο γραφικό παραλληλόγραμμο χρησιμοποιείται σαν φόντο στην περιοχή σχεδίασης. Χρησιμοποιώντας πάλι έναν πίνακα A και την συνάρτηση assign προστίθενται attributes στο αντικείμενο backdrop ώστε να καθοριστεί η εμφάνισή του. Κατόπιν δημιουργούνται και δύο στοιχεία SVG, τύπου SVGpoint, τα οποία αντιπροσωπεύουν σημεία στο δισδιάστατο επίπεδο και χρησιμοποιούνται ως βοηθητικές δομές για αποθήκευση συντεταγμένων που συνδέονται με τα events του κέρσορα του ποντικού.

5.3.2 Υλοποίηση σχεδίασης τοίχων

Όσον αφορά τον σχεδιασμό των κατόψεων αναφέρθηκε και πριν ότι δόθηκε μεγάλη βάση στην υλοποίηση του εργαλείου σχεδίασης τοίχων, καθώς αποτελούν τον σκελετό του σχεδίου πάνω στον οποίο καθορίζονται μετά οι χώροι και τα αντικείμενα. Το σύστημα των τοίχων στο δισδιάστατο σχέδιο είναι πολύ ευέλικτο και οργανικό καθώς, επιτρέπει την σύνδεση τοίχων με κόμβους και την μετακίνηση των κόμβων, με αποτέλεσμα την αλλαγή προσανατολισμού των τοίχων που είναι συνδεδεμένοι στον εκάστοτε κόμβο κτλ. Ο κάθε τοίχος, ξεχωριστά σαν οντότητα, αποτελείται από το σημείο έναρξης και το σημείο τέλους. Η μεταξύ τους απόσταση, αποτελεί το μήκος του τοίχου και ο προσανατολισμός της ευθείας που ενώνει τα δύο σημεία, αποτελεί τον προσανατολισμό του. Τα δύο αυτά σημεία καθορίζουν τα

βασικά χαρακτηριστικά του τοίχου στο σχέδιο. Τα άλλα δύο χαρακτηριστικά είναι το ύψος και το πάχος τα οποία μπορούν εύκολα να καθοριστούν από αριθμητικές παραμέτρους. Η τοποθέτηση του τοίχου στο σχέδιο εξαρτάται από την τοποθέτηση των σημείων έναρξης και τέλους. Διαπιστώθηκε από τα πρώτα στάδια σχεδιασμού της εφαρμογής ότι αυτά τα δύο σημεία θα είναι κρίσιμα για τον καθορισμό πολλών στοιχείων του σχεδίου κατόψεως.

5.3.3 Υλοποίηση αντικειμένων Handle

Σε κώδικα Javascript δημιουργήσαμε την κλάση αντικειμένου Handle, η οποία αντιπροσωπεύει τα «χερούλια», δηλ αυτά τα κομβικά σημεία επιτρέπουν στον χρήστη να ελέγξει τον προσανατολισμό και το μήκος των τοίχων. Η γραφική αναπαράσταση των handles γίνεται με την μορφή μικρών λευκών κύκλων με μαύρο περίγραμμα. Το αρχείο Javascript που υλοποιεί την λογική των Handles είναι το Handle.js και ακολουθεί παρουσίαση του κώδικά του. Η κλάση Handle δηλώνεται ως εξής:

ΚΩΔΙΚΑΣ #2:

```
1.  function Handle(x,y)
2.  {
3.        this.id = Handles.getNextId();

4.         // primitive data
5.         this.x = x;
6.         this.y = y;

7.         // SVG reference
8.         this.SVG = null;

9.         // refernces to other objects handled
10. this.walls = new Array();
11. this.floors = new Array();
12. this.fillArray = new Array();

13. //actions

14. //add to array
15. Handles.addHandle(this);

16. //draw the element on screen
17. this.draw();
18. }
```

Παρατηρούμε καταρχήν ότι η συνάρτηση κατασκευής δέχεται δύο ορίσματα x και y, τα οποία αντιπροσωπεύουν τα σημεία του επιπέδου σχεδίασης, στα οποία θα οριστεί να δημιουργηθεί ένα νέο handle. Τα handles μπορούν να δημιουργηθούν καλώντας με το keyword new την συνάρτηση Handle π.χ.:

```
myHandle = new Handle(30,30);
```

Η παραπάνω εντολή δημιουργεί ένα αντικείμενο Handle, το οποίο αντιπροσωπεύεται γραφικά από έναν μικρό κύκλο χρώματος λευκού και περιγράμματος μαύρου ο οποίος έχει κέντρο το σημείο 30,30 του επιπέδου.

Παρατηρώντας την συνάρτηση κατασκευής του αντικειμένου Handle βλέπουμε ότι έχει κάποιες ιδιότητες/attributes όπως:

- **Id:** ένα αλφαριθμητικό το οποίο χρησιμοποιείται για αναφορά στο συγκεκριμένο αντικείμενο και χρησιμοποιείται πάρα πολύ στο serialization των δομών κατά την αποθήκευση του σχεδίου σε αρχείο στον δίσκο.
- **x:** η συντεταγμένη x του σημείου στο οποίο βρίσκεται το handle. Παρατηρούμε ότι (κώδικας #2, γραμμή #5) η ιδιότητα x εξισώνεται με το όρισμα x που διοχετεύεται κατά την κλήση κατασκευής του αντικειμένου:
  ```
  this.x = x;
  ```

- **y:** η συντεταγμένη y του σημείου στο οποίο βρίσκεται το handle. Παρατηρούμε ότι (κώδικας #2, γραμμή #6) η ιδιότητα y εξισώνεται με το όρισμα y που διοχετεύεται κατά την κλήση κατασκευής του αντικειμένου:
  ```
  this.y = y;
  ```

- **SVG:** Αποτελεί αναφορά στο γραφικό στοιχείο SVG το οποίο αποτελεί την οπτική αναπαράσταση του αντικειμένου handle στην οθόνη του χρήστη.
- **walls:** Αποτελεί έναν πίνακα στον οποίο θα τοποθετηθούν αναφορές στα αντικείμενα των τοίχων που συνδέονται σε αυτό το handle.
- **floors:** Αποτελεί έναν πίνακα στον οποίο θα τοποθετηθούν αναφορές στα αντικείμενα των δαπέδων που οριοθετούνται από αυτό το handle.
- **fillArea:** Αποτελεί έναν πίνακα που χρησιμοποιείται βοηθητικά για να οριστεί η περιοχή η οποία δημιουργείται στις άκρες συνένωσης των τοίχων (περιγράφεται με λεπτομέρειες παρακάτω στο κεφάλαιο αυτό).

Βλέπουμε ότι κατά την δημιουργία έχουμε κλήση της μεθόδου του αντικειμένου Handles.addHandle(this). Το Handles αποτελεί ένα αντικείμενο (τύπου HandleArray που θα περιγραφεί παρακάτω), το οποίο κρατάει ένα array από όλα τα handles που έχουν δημιουργηθεί κατά την εκτέλεση του προγράμματος, ώστε να υπάρχει μια γενική εποπτεία αυτών και δυνατότητα αναφοράς. Το αντικείμενο τύπου Handle περιέχει αρκετές μεθόδους μερικές από τις οποίες αναφέρονται παρακάτω.

Υλοποιείται η drawHandleShape() η οποία είναι υπεύθυνη για την δημιουργία όλων εκείνων των SVG στοιχείων που αποτελούν την γραφική αναπαράσταση του αντικειμένου handle στην περιοχή σχεδίασης:

ΚΩΔΙΚΑΣ #3:

```
1.  function drawHandleShape(id,x,y,r,fill,stroke,sw)
2.  {

3.  var new_circle = document.createElementNS(xmlns,"circle");

4.  var A =
5.  {
6.  "type":"handle",
7.  "id":id,
8.  "cx":"0",
9.  "cy":"0",
10.       "r":r,
11.       "fill":fill,
```

```
12.       "stroke":stroke,
13.       "stroke-width": sw,
14.       "transform":"translate(" + x + "," + y + ")"

15.       }
16.       assign(new_circle,A);
17.       Handles.group.appendChild(new_circle);
18.       return new_circle;

19.       }
```

Παρατηρούμε ότι η συγκεκριμένη συνάρτηση παίρνει κάποια ορίσματα παραμέτρους τα οποία καθορίζουν τις ιδιότητες των στοιχείων SVG που θα δημιουργηθούν. Όπως αναφέραμε το αντικείμενο handle αναπαρίσταται από έναν κύκλο, άρα αρκεί να δημιουργηθεί ένα αντικείμενο SVG τύπου circle το οποίο και γίνεται (κώδικας #3, γραμμή #3).

```
var new_circle = document.createElementNS(xmlns,"circle");
```

Η εντολή createElementNS, που περιγράφηκε και προηγουμένως, χρησιμοποιείται για την δημιουργία νέων στοιχείων στην ιστοσελίδα. Κατόπιν χρησιμοποιώντας ένα array A, δηλώνονται όλα τα ζεύγη attribute – value που θα χαρακτηρίσουν το αντικείμενο και γίνεται κλήση της συνάρτησης assign. Από τις ιδιότητες που περιγράφονται στον πίνακα A παρατηρούμε ότι το στοιχείο circle έχει κέντρο το 0,0 (attributes cx=0 και cy=0) αλλά δέχεται ένα attribute transform με τιμή translate(x,y) το οποίο το μετακινεί στις συντεταγμένες του σημείου στο οποίο ορίζεται το handle. Τέλος η συνάρτηση επιστρέφει αναφορά στο SVG element circle. Η συνάρτηση αυτή χρησιμοποιείται από την μέθοδο draw του αντικειμένου Handle. Παρατηρούμε ότι κάθε μέθοδος αντικειμένου στην Javascript υλοποιείται με την χρήση prototypes.

ΚΩΔΙΚΑΣ #4:

```
1.  Handle.prototype.draw = function()
2.  {
3.  if (this.SVG!=null) return;
4.  this.SVG = drawHandleShape(this.id,this.x,this.y,7,"white","black",2);

5.  }
```

Κατά την κλήση της συνάρτησης κατασκευής του αντικειμένου Handle η ιδιότητα SVG παραμένει κενή. Στο τέλος της συνάρτησης κατασκευής γίνεται κλήση της μεθόδου draw η οποία ελέγχει αν η ιδιότητα SVG είναι κενή. Αν είναι κενή, σημαίνει ότι η γραφική αναπαράσταση του αντικειμένου με στοιχεία SVG δεν υφίσταται ακόμα, οπότε καλείται η drawHandleShape() η οποία δημιουργεί τα κατάλληλα SVG

elements (στην συγκεκριμένη περίπτωση έναν κύκλο) και επιστρέφει την αναφορά στην ιδιότητα SVG. Άρα μέσω κώδικα δίνεται η δυνατότητα να αλλάξουμε την γραφική αναπαράσταση του αντικειμένου Handle μπορούμε μέσω του property Handle.SVG. Το property αυτό μας δίνει πρόσβαση στο στοιχείο SVG που αποτελεί την γραφική αναπαράσταση του Handle. Αντίστοιχα υπάρχει και η μέθοδος undrawn() η οποία «καταστρέφει» την γραφική αναπαράσταση του αντικειμένου Handle στην περιοχή σχεδίασης:

ΚΩΔΙΚΑΣ #5:

```
1. Handle.prototype.undraw = function()
2. {
3. if (this.SVG != null)
4. {
5. this.SVG.parentNode.removeChild(this.SVG);
6. this.SVG = null;
7. }
8. }
```

Η συνάρτηση undrawn() ελέγχει πρώτα αν υφίσταται η γραφική αναπαράσταση του αντικειμένου (ελέγχοντας αν η ιδιότητα Handle.SVG είναι διάφορη του null – Κώδικας #5, Γραμμή #3). Αν όντως είναι διάφορη του null, σημαίνει ότι η ιδιότητα.SVG περιλαμβάνει μια αναφορά σε αντικείμενο SVG το οποίο πρέπει να διαγραφεί. Η διαγραφή στην ουσία γίνεται από το SVG DOM. Χρησιμοποιώντας την ιδιότητα parentNode επισκεπτόμαστε τον γονέα του αντικειμένου SVG <circle> και του επιβάλουμε να διαγράψει το παιδί <circle>. Στην ουσία διαγράφεται η αναφορά του γονέα προς το παιδί στην δενδρική δομή του Document Object Model και ο garbage collector της Javascript διαγράφει αυτόματα τα «ξεκρέμαστα», χωρίς αναφορές, αντικείμενα της γλώσσας και του περιβάλλοντος του φυλλομετρητή (Κώδικας #5, γραμμή #5). Παράλληλα με την κλάση Handle υλοποιείται και η κλάση HandleArray η οποία κρατάει στην ουσία μια λίστα με αναφορές σε όλα τα αντικείμενα Handle που έχουν δημιουργηθεί μέσα στο περιβάλλον σχεδίασης της εφαρμογής. Η δήλωση της HandleArray σαν κλάση αντικειμένων γίνεται στο ίδιο αρχείο Handle.js:

ΚΩΔΙΚΑΣ #6:

```
1. function HandleArray() {
2. this.id = 0;
3. this.list = new Array();
4. this.group = createHandleGroup();
5. }
```

Παρατηρώντας τον παραπάνω κώδικα βλέπουμε ότι το αντικείμενο της κλάσης HandleArray έχει 3 βασικές ιδιότητες:

- **Id:** αποτελεί έναν μετρητή αύξοντα αριθμού ο οποίος χρησιμοποιείται σαν μήτρα για την απόδοση id στα νέα αντικείμενα τύπου Handle που δημιουργούνται.
- **List:** μια μονοδιάστατη array (ουσιαστικά μια λίστα) με αναφορές σε όλα τα αντικείμενα Handle που υφίστανται στην εφαρμογή.
- **Group:** αναφορά σε SVG στοιχείο τύπου <g> το οποίο στην ουσία ομαδοποιεί σε ένα σύνολο όλες τις γραφικές αναπαραστάσεις των αντικειμένων Handle. Αυτό γίνεται για να είναι δυνατή ή άμεση εφαρμογή γραφικών μεθόδων σε όλα τα γραφικά στοιχεία τύπου Handle όπως π.χ. καθολική εμφάνιση ή απόκρυψη από την περιοχή σχεδίασης.

Η συνάρτηση createHandleGroup() (γραμμή 4 κώδικα 6) χρησιμοποιείται για να δημιουργήσει το SVG στοιχείο <g>, το οποίο αποτελεί container για όλα τα SVG στοιχεία τύπου <circle> που αποτελούν γραφικές αναπαραστάσεις των αντικειμένων τύπου Handle. Ο κώδικας της createHandleGroup είναι ο εξής:

ΚΩΔΙΚΑΣ #7:

```
1. function createHandleGroup()
2. {
3. var new_grp = document.createElementNS(xmlns,"g");
4. var A =
5. {
6. "id":"handleGroup"
7. }
8. assign(new_grp,A);
9. SVGroot.appendChild(new_grp);
10. return new_grp;
11. }
```

Όπως παρατηρούμε (κώδικας #7, γραμμή #3) υπάρχει κλήση, της γνωστής πλέον μεθόδου createElementNS, η οποία δέχεται φυσικά το SVG namespace μέσω της global μεταβλητής xmlns. Επίσης διοχετεύεται ο τύπος του αντικειμένου προς δημιουργία ο οποίος είναι το “g” διότι πρόκειται να δημιουργηθεί ένα SVG group element <g>.

Παρακάτω αναφέρονται μερικές από τις μεθόδους της κλάσης HandleArray:

i) getNextId()
Πρόκειται για μια μέθοδο η οποία επιστρέφει ένα νέο διακριτικό id για το προς κατασκευή αντικείμενο handle.

ii) Μέθοδοι addHandle() και removeHandle()
Χρησιμοποιούνται για την προσθήκη/διαγραφή αντικειμένων handle από την διάταξη list του HandleArray.

iii) Μέθοδοι hide() / show()
Αρκετές φορές στην εφαρμογή είναι απαραίτητο στην περιοχή σχεδίασης να αποκρύπτονται καθολικά όλα τα Handles (που αναπαρίστανται γραφικά με κύκλους)

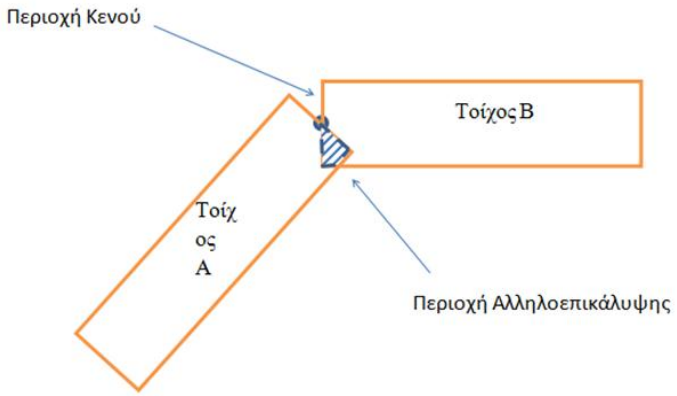
και να επανεμφανίζονται. Αυτό επιτυγχάνεται με τις δύο μεθόδους hide() και show() του αντικειμένου HandleArray.

iv) Μέθοδοι getHandle() και getHandleById()
Οι δύο μέθοδοι χρησιμοποιούνται για να βρούμε συγκεκριμένα objects τύπου handle όταν γνωρίζουμε το SVG element που τα αναπαριστά γραφικά ή όταν γνωρίζουμε το id τους.

5.3.4. Υλοποίηση αντικειμένων Wall

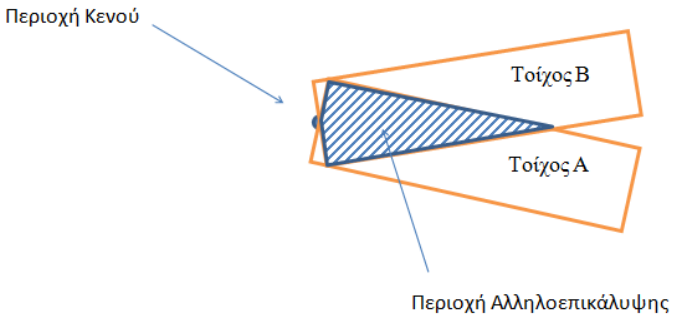
Αφού έγινε αναλυτική περιγραφή των αντικειμένων Handles, τα οποία χρησιμοποιούνται ως βοηθητικά στοιχεία ελέγχου για τον καθορισμό άλλων βασικών στοιχείων των σχεδίων όπως τοίχοι και δωμάτια, θα αναφερθούμε τώρα στην δομή η οποία χρησιμοποιείται για την περιγραφή των τοίχων ως λογικά object. Η δομή αυτή είναι η κλάση αντικειμένων Wall. Όλες οι ιδιότητες και η μέθοδοι της κλάσης Wall, περιγράφονται στο αρχείο Javascript wall.js. Πριν γίνει περιγραφή του κώδικα που υλοποιεί την κλάση αντικειμένων wall θα πρέπει να αναφερθούν τα στάδια σχεδιασμού της υλοποίησης της συγκεκριμένης δομής, ώστε να γίνει κατανοητή η τελική μορφή της στον κώδικα. Κατά τον αρχικό σχεδιασμό θεωρήθηκε ότι ένα αντικείμενο τοίχου περιγράφεται από δύο σημεία: Το σημείο έναρξης (start) και το σημείο λήξης (end). Στην ουσία ο τοίχος είναι το ευθύγραμμο τμήμα που περιέχεται μεταξύ των δύο σημείων. Άρα για την πλήρη τοποθέτησή του στο επίπεδο απαιτούνται (σαν πληροφορία) δύο ζεύγη συντεταγμένων (σημείο έναρξης και λήξης). Για την πλήρη περιγραφή του χρειάζονται και άλλες δύο αριθμητικές παραμέτρους οι οποίες είναι το ύψος και το πάχος. Για τον δυναμικά ελεγχόμενο ορισμό των σημείων στο επίπεδο χρησιμοποιείται η κλάση handle. Άρα η κλάση wall θα πρέπει περιγράφει το ευθύγραμμο τμήμα μεταξύ των δύο σημείων. Το ευθύγραμμο τμήμα αυτό έχει και ένα συγκεκριμένο πάχος (η παράμετρος του ύψους δεν παίζει ρόλο στην δισδιάστατη απεικόνιση). Χρησιμοποιώντας τεχνολογία SVG επιλέχθηκε αρχικά, ο τοίχος γραφικά να αναπαρασταθεί με ένα ευθύγραμμο τμήμα με μεταβλητό πάχος. Στα γραφικά SVG υπάρχει συγκεκριμένο element το οποίο περιγράφει την απεικόνιση ενός ευθύγραμμου τμήματος και είναι το SVG element <line>, το οποίο φυσικά έχει τα εξής βασικά attributes που καθορίζουν την θέση του στο επίπεδο: x1,y1,x2,y2. Τα attributes αυτά αναφέρονται φυσικά στις συντεταγμένες δύο σημείων (x1,y1) και (x2,y2) που οριοθετούν το τμήμα. Επίσης χρησιμοποιώντας την ιδιότητα stroke-width αλλάζει το πάχος του στοιχείου <line> ώστε να ανταποκρίνεται στο πάχος του εκάστοτε τοίχου. Κοιτώντας όμως, θεωρητικά μπροστά στις απαιτήσεις του project, διαπιστώθηκε πως θα ήταν επιθυμητό στο δισδιάστατο σχέδιο να αναπαρίσταται γραφικά η υφή (ταπετσαρία), του εκάστοτε τοίχου. Για να συμβεί αυτό θα πρέπει ο τοίχος να αναπαριστάται από ένα στοιχείο SVG το οποίο να έχει ιδιότητα fill. Έτσι επιλέχθηκε να αναπαρασταθεί γραφικά ο κάθε τοίχος με ένα ορθογώνιο το οποίο σε SVG αντιστοιχεί στο στοιχείο <rect>. Φυσικά τα attributes του στοιχείου <rect> είναι x,y,width,height με λίγα λόγια

μπορούν να αναπαραστήσουν οριζόντια παραλληλόγραμμα με συγκεκριμένο σημείο έναρξης και κατόπιν συγκεκριμένο ύψος και πλάτος. Επειδή όμως οι τοίχοι στο σχέδιο έχουν προσανατολισμό θα πρέπει το στοιχείο <rect> να τοποθετηθεί σε ένα στοιχείο container τύπου <g> στο οποίο να αποδοθεί μετασχηματισμός, με κατάλληλη μετατόπιση και περιστροφή, ώστε να μπορεί το <rect> να αναπαριστά έναν τοίχο ο οποίος μπορεί να βρίσκεται οπουδήποτε στο επίπεδο και επίσης να έχει οποιαδήποτε γωνία για προσανατολισμό. Αυτή η επιλογή φάνηκε επαρκής για την πλήρη αναπαράσταση του τοίχου. Κατά την προσθήκη, όμως, επιπλέων τοίχων στο σχέδιο και μάλιστα σε σύνδεση μεταξύ τους, δηλ. τοίχους που συνδέονταν και δημιουργούσαν γωνίες, παρατηρήθηκε ότι τα ορθογώνια παραλληλόγραμμα δημιουργούσαν πρόβλημα στους κόμβους καθώς αλληλεπικάλυπταν το ένα το άλλο ή αφήνανε κενά στις γωνίες. Τα προβλήματα αυτά αναπαριστώνται στην παρακάτω εικόνα:



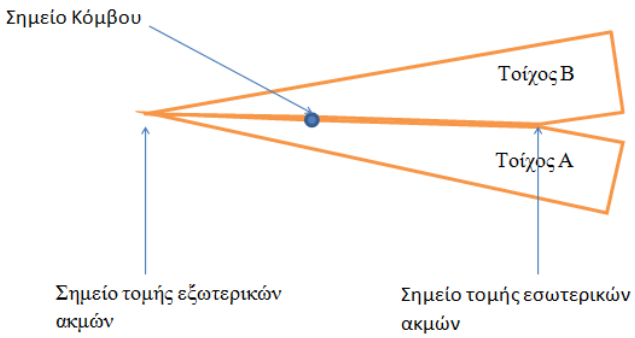
Εικόνα 78 - Απεικόνιση των προβλημάτων στις συνδέσεις των τοίχων.

Επίσης όσο πιο κλειστή γωνία σχημάτιζαν δύο τοίχοι μεταξύ τους τόσο πιο μεγάλη ήταν η επιφάνεια αλληλοεπικάλυψης στην εσωτερική γωνία και τόσο πιο μεγάλο το κενό στην εξωτερική.



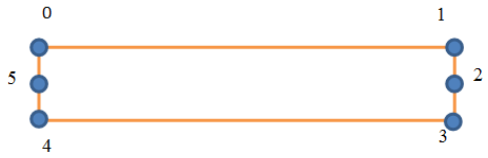
Εικόνα 79 - Περιγραφή των προβλημάτων σύνδεσης μεταξύ τοίχων.

Κανονικά οι τοίχοι κατά την σύνδεσή τους θα πρέπει να έχουν την εξής μορφή:



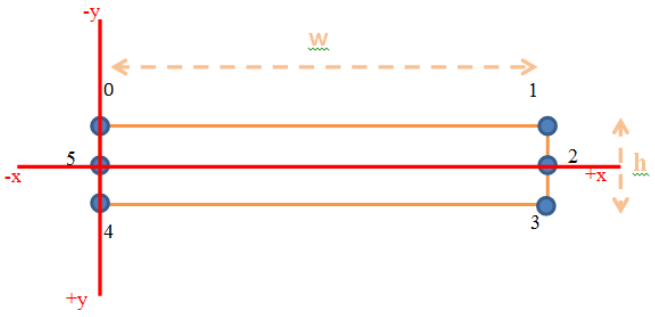
Εικόνα 80 - Απεικόνιση της σωστής σύνδεσης τοίχων.

Για να αντιμετωπιστούν τα προβλήματα σύνδεσης οι τοίχοι θα πρέπει να αναπαριστούνται από πολύγωνα. Στην γλώσσα SVG υπάρχει αντίστοιχο στοιχείο <polygon> το οποίο περιγράφει ένα πολυγωνικό σχήμα. Συγκεκριμένα σύμφωνα με την μελέτη που έγινε για την γεωμετρική αναπαράσταση της σύνδεσης των τοίχων κάθε τοίχος θα πρέπει να περιγράφεται από ένα πολύγωνο με 6 σημεία. Το πολύγωνο περιγράφεται στην παρακάτω εικόνα:



Εικόνα 81– Απεικόνιση των σημείων από τα οποία αποτελείται το πολύγωνο τοίχου.

Παρατηρούμε ότι αν και είναι πολύγωνο οι πλευρές του είναι έτσι τοποθετημένες που διατηρεί την μορφή ενός ορθογωνίου. Οι κόμβοι του πολυγώνου είναι αριθμημένοι με την σειρά. Ξαναπαρουσιάζεται η παραπάνω εικόνα προσθέτοντας το σύστημα συντεταγμένων που αντιστοιχεί στο SVG στοιχείο <polygon> κατά την δήλωσή του:



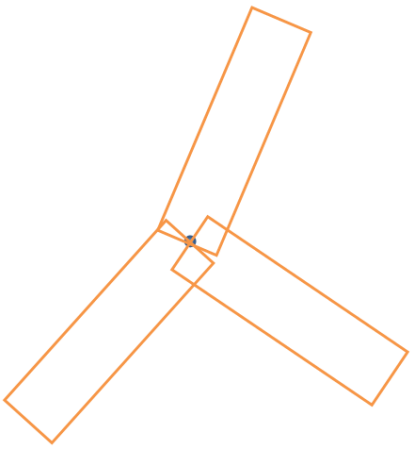
Εικόνα 82 - Απεικόνιση του συστήματος συντεταγμένων του πολυγώνου του τοίχου.

Έστω σύμφωνα με την παραπάνω εικόνα ότι το μήκος του τοίχου είναι w και το πάχος του είναι h. Το σημείο 5 συμπίπτει με το σημείο 0,0 των αξόνων. Επίσης το σημείο (2) του πολυγώνου έχει συντεταγμένες (w,0). Το σημείο (5) παραμένει σταθερό ενώ, το σημείο (2) αλλάζει ως προς την συντεταγμένη x, εφόσον αλλάξει το

μήκος του τοίχου. Τα δύο αυτά σημεία του πολυγώνου είναι σημαντικά γιατί εκεί πάντα θα βρίσκονται οι κόμβοι σύνδεσης με άλλους τοίχους. Το σημείο (0) έχει συντεταγμένες (0,-h/2) (όπου h/2 το μισό του πάχους του τοίχου). Το σημείο (1) έχει συντεταγμένες (w,-h/2) ενώ το (3) (w,h/2) και το (4) (0,h/2). Έτσι βλέπουμε πως όλα τα σημεία του πολυγώνου εκφράζονται συναρτήσει των παραμέτρων μήκους και πάχους του τοίχου. Όταν στον συγκεκριμένο τοίχο συνδεθεί άλλος τοίχος από τα αριστερά τότε ο κόμβος σύνδεσης θα συμπίπτει με το σημείο (0) του πολυγώνου της εικόνας. Ανάλογα τον προσανατολισμό και το πάχος του άλλου τοίχου τα σημεία (0) και (4) του πολυγώνου θα μετακινηθούν ώστε να μην παρουσιάζεται περιοχή αλληλοεπικάλυψης ούτε περιοχή κενού κατά την σύνδεση. Για το πώς μετακινούνται κατάλληλα αυτά τα σημεία των πολυγώνων μελετήσαμε αρκετές περιπτώσεις συνδέσεως τοίχων και καταλήξαμε σε συγκεκριμένο αλγόριθμο που καλύπτει όλες τις περιπτώσεις.

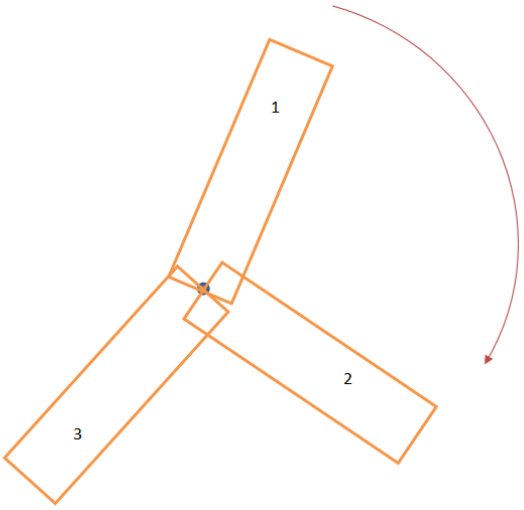
5.3.5 Διαδικασία εξομάλυνσης πολυγώνων στα σημεία σύνδεσης των τοίχων

Η συγκεκριμένη διαδικασία λειτουργεί ορθά για κάθε αριθμό τοίχων που συνδέονται σε έναν κόμβο. Σκοπός της είναι να εξομαλύνει τα πολύγωνα που αναπαριστούν τους τοίχους ώστε να μην υπάρχουν επικαλύψεις ή κενά στους κόμβους σύνδεσης. Έστω ότι σε έναν κόμβο συνδέονται τρείς τοίχοι όπως στο παρακάτω σχήμα:



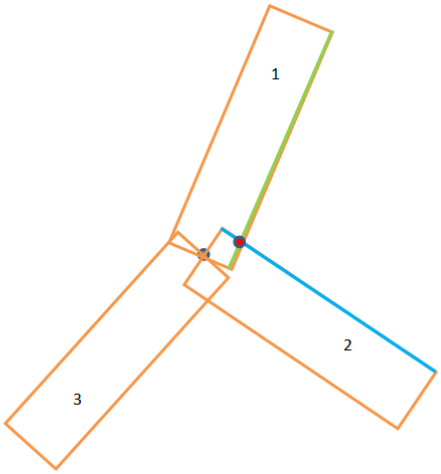
Εικόνα 83 – Παράδειγμα σύνδεσης τριών τοίχων σε κοινό κόμβο.

Με κέντρο τον κόμβο σύνδεσης θεωρείται νοητά ένας κύκλος που περικλείει και τα 3 πολύγωνα των τοίχων. Οι τοίχοι αριθμούνται με την φορά του ρολογιού.



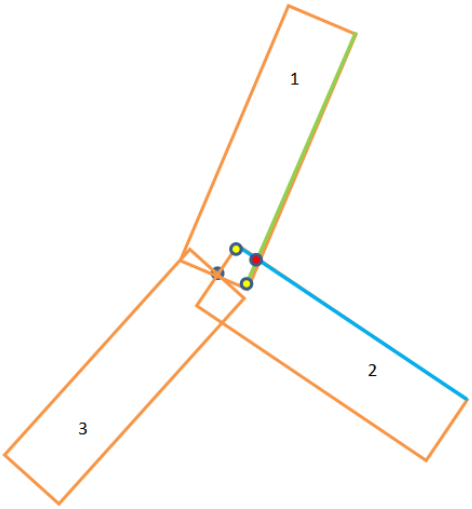
Εικόνα 84 – Έναρξη διαδικασίας με αρίθμηση των τριών τοίχων.

Επιλέγονται κατά ζεύγη οι τοίχους ξεκινώντας από τον πρώτο και γίνεται υπολογισμός του σημείο τομής της «δεξιάς» πλευράς του πρώτου με την «αριστερή πλευρά» του δεύτερου. Σημειώνονται με τα αντίστοιχα χρώματα στο παρακάτω σχήμα ενώ το σημείο τομής τους σημειώνεται με κόκκινο:



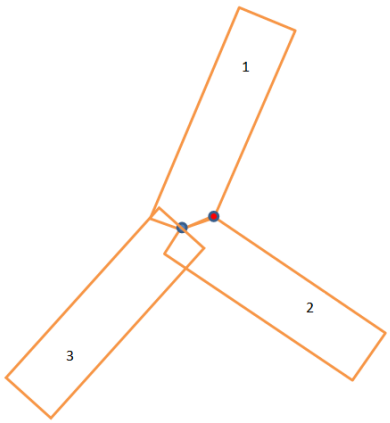
Εικόνα 85 – Εύρεση σημείου τομής μεταξύ των δύο τοίχων.

Στο σημείο τομής θα πρέπει να μετακινηθούν τα συγκεκριμένα σημεία των πολυγώνων 1 και 2 τα οποία εμφανίζονται με κίτρινο χρώμα στο σχήμα:



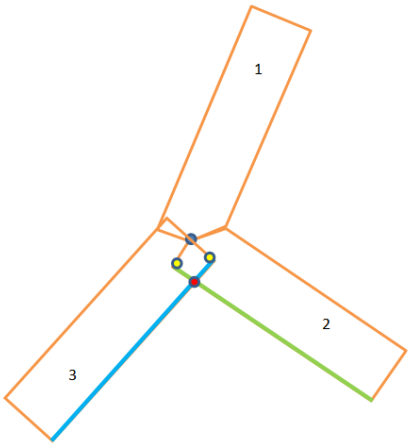
Εικόνα 86 – Επισήμανση των σημείων που θα μετακινηθούν στο σημείο τομής.

Τα δύο σημεία που επισημαίνονται με κίτρινο χρώμα θα ταυτιστούν με το σημείο τομής που επισημαίνεται με κόκκινο. Μετά την μετακίνησή τους το σχήμα θα έχει ως εξής:



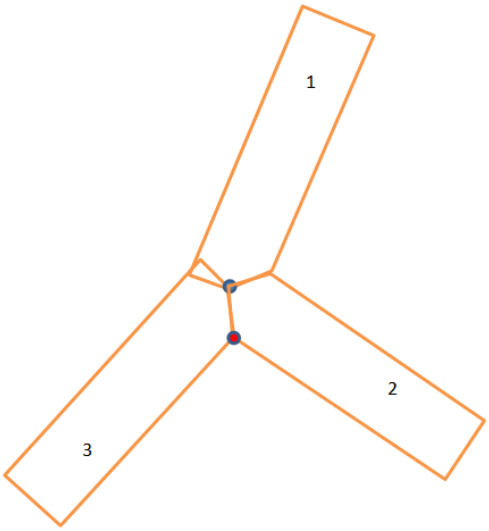
Εικόνα 87 – Μετακίνηση στο σημείο τομής.

Η διαδικασία επαναλαμβάνεται για το ζεύγος τοίχων 2 και 3. Υπολογίζεται το σημείο τομής μεταξύ της «δεξιάς» πλευράς του τοίχου 2 και της «αριστερής» πλευράς του τοίχου 3. Επίσης επισημαίνονται με κίτρινο χρώμα τα σημεία των πολυγώνων που θα πρέπει να μετακινηθούν και να ταυτιστούν με το σημείο τομής.



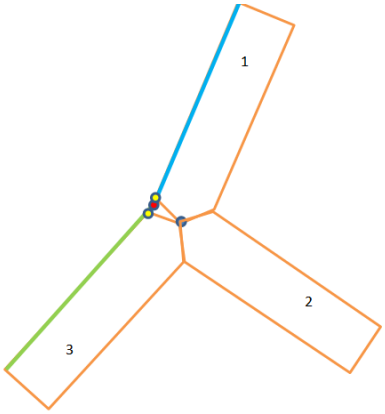
Εικόνα 88 – Εύρεση του επόμενου σημείου τομής.

Μετά την μετακίνηση των δύο σημείων το σχήμα έχει ως εξής:



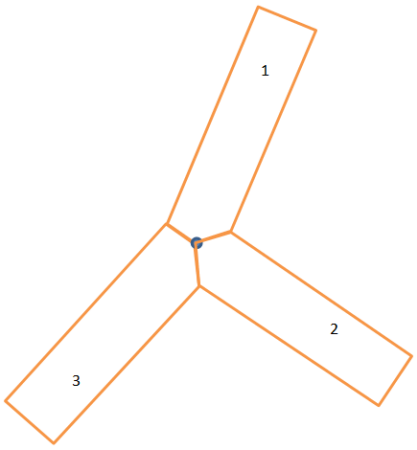
Εικόνα 89 – Εξομάλυνση των πολυγώνων των τοίχων

Η διαδικασία επαναλαμβάνεται για όλα τα ζεύγη τοίχων μέχρι να φτάσουμε στον ζεύγος τελικού τοίχου με αρχικό. Στην συγκεκριμένη περίπτωση, στο επόμενο βήμα ο αλγόριθμος επιλέγει τον τοίχο 3 και τον τοίχο 1 σαν τελικό στάδιο. Κατά την επιλογή τους εκτελεί πάλι τα ίδια βήματα όπως και με τα προηγούμενα ζεύγη. Έτσι υπολογίζεται το σημείο τομής μεταξύ των πλευρών και τα σημεία των πολυγώνων που πρέπει να μετακινηθούν σύμφωνα με το παρακάτω σχήμα:



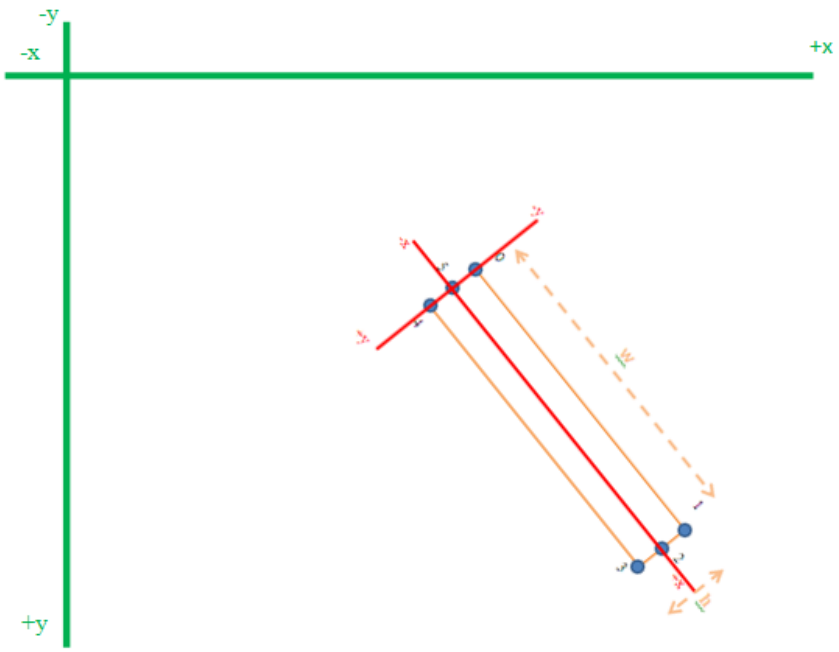
Εικόνα 90 – Εύρεση τελικού σημείου τομής.

Μετά την μετακίνησή τους η διαδικασία τερματίζει και το σχήμα έχει ως εξής:



Εικόνα 91 – Ολοκλήρωση της διαδικασίας εξομάλυνσης.

Η υλοποίηση της διαδικασίας θα περιγραφεί αναλυτικότερα σε επίπεδο κώδικα αργότερα στο κεφάλαιο αυτό. Κάθε τοίχος λοιπόν στην εφαρμογή, αναπαριστάται από πολύγωνο 6 σημείων. Τα σημεία αυτά αριθμούνται από το (0) έως το (5). Τα σημεία (0) και (2) αποτελούν τα σημεία τα οποία οριοθετούν τον τοίχο. Στα σημεία αυτά εμφανίζονται και τα Handles τα οποία δίνουν την δυνατότητα στον χρήστη να αλλάξει το μέγεθος και τον προσανατολισμό του τοίχου. Για να δημιουργηθεί λοιπόν ένας τοίχος προαπαιτείται η ύπαρξη δύο Handles για να τον οριοθετούν. Τα Handles αποτελούν τα σημεία κόμβων στους οποίους συνδέονται οι τοίχοι. Επίσης το πολύγωνο κάθε τοίχου τοποθετείται μέσα σε ένα στοιχείο <g> στο οποίο με εφαρμόζονται κατάλληλοι γεωμετρικοί μετασχηματισμοί. Αυτό σημαίνει ότι μέσα στο <g>, το σύστημα συντεταγμένων του στοιχείου <polygon> διατηρείται. Το σύστημα συντεταγμένων του στοιχείου <polygon> σε σχέση με το αρχικό σύστημα συντεταγμένων λόγω του μετασχηματισμού του στοιχείου <g> αναπαρίσταται στο παρακάτω σχήμα:



Εικόνα 92 – Το εμφωλευμένο σύστημα συντεταγμένων ενός πολύγονου που αναπαριστά τοίχο.

5.3.6 Υλοποίηση των αντικειμένων τύπου Wall

Ακολουθεί το τμήμα του κώδικα στο οποίο έχουμε την δήλωση της κλάσης αντικειμένων Wall η οποία χρησιμοποιείται για την δημιουργία αντικειμένων που περιγράφουν τους τοίχους που περιέχει το σχέδιο:

ΚΩΔΙΚΑΣ #8:

```
1.  function Wall(_h1,_h2)
2.  {

3.  this.id = Walls.getNextId();

4.  this.SVG = null;
5.  this.poly = null;
6.  this.start = null;
7.  this.end = null;

8.  this.windows = new Array();
9.  this.height = 120;
10. this.depth = 20;
11. this.typeA = "paint";
12. this.typeB = "paint";
13. this.colorA = new Color(155,155,155);
14. this.colorB = new Color(155,155,155);
15. this.textureA = null;
16. this.textureB = null;

17. this.colorA.getColor(Walls.colorA);
18. this.colorB.getColor(Walls.colorB);
```

```
19. this.type = "concrete";

20. Walls.addWall(this);

21. if (_h1 != null && _h2 != null)
22. {
23. this.setStart(_h1);
24. this.setEnd(_h2);
25. this.draw();
26. this.updateSides();
27. _h1.calibrateWalls();
28. _h2.calibrateWalls();
29. }
30. }
```

Η συνάρτηση κατασκευής του αντικειμένου Wall δέχεται δύο ορίσματα _h1 και h2 τύπου Handle. Αυτό συμβαίνει γιατί όπως προαναφέραμε κάθε αντικείμενο Wall στην εφαρμογή μας απαιτεί την ύπαρξη δύο Handle objects για να οριστεί. Παρατηρούμε ότι η κλάση Wall έχει αρκετά attributes τα πιο σημαντικά εκ των οποίων είναι :

- **Id:** όπως και στην κλάση handle έτσι και εδώ κάθε αντικείμενο που δημιουργείται παίρνει ένα ξεχωριστό id.
- **SVG:** αναφορά στο SVG element <g> που περιέχει τα στοιχεία γραφικών που αναπαριστούν τον τοίχο.
- **Poly:** αναφέρεται στο πολύγωνο που αναπαριστά τον τοίχο.
- **Start:** αναφορά στο handle που οριοθετεί την έναρξη του τοίχου (συμπίπτει με το σημείο 0 του πολύγωνου).
- **End:** αναφορά στο handle που οριοθετεί την λήξη του τοίχου (συμπίπτει με το σημείο 2 του πολύγωνου).
- **Windows:** Ένα array το οποίο περιλαμβάνει αναφορές σε όλα τα στοιχεία οπές του τοίχου (δηλαδή παράθυρα, πόρτες, οπές).
- **Height:** το ύψος του τοίχου.
- **Depth:** το πάχος του τοίχου.
- **TypeA:** αναφέρεται στον τύπο (χρώμα ή υφή) με τον οποίο είναι διακοσμημένη η πρώτη πλευρά του τοίχου.
- **TypeB:** τύπος (χρώμα ή υφή) διακόσμησης της δεύτερης πλευράς του τοίχου.
- **colorA:** αν ο τύπος διακόσμησης της πρώτης πλευράς είναι χρώμα σε αυτή την attribute αποθηκεύονται πληροφορίες για το χρώμα.
- **colorB:** το ίδιο με το παραπάνω αλλά για την δεύτερη πλευρά.
- **textureA:** αναφορά στην υφή που διακοσμεί την πρώτη πλευρά του τοίχου.
- **textureB:** αναφορά στην υφή που διακοσμεί την δεύτερη πλευρά του τοίχου.
- **type:** Ο τύπος κατασκευής του τοίχου.

Επίσης όπως και με την κλάση Handle και Handle Array και εδώ υπάρχει η κλάση WallArray, η οποία μοιάζει πάρα πολύ με την HandleArray και ο σκοπός της είναι να διατηρεί μια λίστα αναφορών στα αντικείμενα wall που υφίστανται στην εφαρμογή. Η κλάση Wall Array έχει κάποιες παρόμοιες μεθόδους με την κλάση HandleArray. Η συνάρτηση createWallShape() καλείται για να δημιουργήσει όλα εκείνα τα στοιχεία SVG τα οποία είναι απαραίτητα για την σωστή γραφική αναπαράσταση του εκάστοτε Wall Object. Ο κώδικας της συνάρτησης παρουσιάζεται παρακάτω:

ΚΩΔΙΚΑΣ #9:

```
1. function createWallShape(w,id,x1,y1,x2,y2,depth)
2. {
3. var poly = document.createElementNS(xmlns,"polygon")
4. var width = calcDistance(x1,y1,x2,y2);
5. var angle = calcAngle(x1,y1,x2,y2);
6. // the wall is essentially a polygon with 6 points integral to our
   implementation
7. poly.points.appendItem(createPoint(0,0));
8. poly.points.appendItem(createPoint(0,-(depth/2)));
9. poly.points.appendItem(createPoint(width,-(depth/2)));
10. poly.points.appendItem(createPoint(width,0));
11. poly.points.appendItem(createPoint(width,(depth/2)));
12. poly.points.appendItem(createPoint(0,(depth/2)));
13. var A = {
14. "fill":"white",
15. "stroke":"black"
16. }
17. assign(poly,A)
18. var polygroup = document.createElementNS(xmlns,"g");
19. polygroup.appendChild(poly);
20. var new_group = document.createElementNS(xmlns,"g");
21. A = {
22. "id":id,
23. "type":"wall",
24. "transform":"translate(" + x1 + " " + y1 + ") rotate(" + angle + ")"
25. }
26. assign(new_group,A);
27. new_group.appendChild(polygroup);
28. Walls.group.appendChild(new_group);
29. w.SVG = new_group;
30. w.poly = poly;
31. }
```

Η συνάρτηση createWallShape(), δέχεται έναν αριθμό από ορίσματα, όπως το id του Wall object ώστε να τοποθετηθεί αντίστοιχα και στο SVG element επίσης διοχετεύεται σαν όρισμα και το τρέχων αντικείμενο Wall καθώς επίσης και οι συντεταγμένες των start και end Handle Objects με την μορφή x1,y1,x2,y2 και τέλος το έλος το πάχος του τοίχου (depth). Αυτές οι 5 τελευταίες αριθμητικές παράμετροι αρκούν για την γραφική αναπαράσταση του τοίχου στον φυλλομετρητή χρησιμοποιώντας στοιχεία SVG. Καταρχήν θα πρέπει να υπολογιστεί το μήκος του τοίχου. Από γεωμετρία ότι επειδή ο τοίχος οριοθετείται από δύο σημεία και συμπίπτει με το ευθύγραμμο τμήμα ανάμεσά τους το μήκος του θα συμπίπτει με το μήκος του

ευθύγραμμου τμήματος που ενώνει τα δύο σημεία. Έτσι στον παραπάνω κώδικα γραμμή 4 καλείται η βοηθητική συνάρτηση calcDistance() η οποία υπολογίζει την απόσταση μεταξύ δύο σημείων στο επίπεδο. Η calcDistance() περιέχεται στο αρχείο utilities.js και ο κώδικάς που την υλοποιεί είναι ο εξής:

ΚΩΔΙΚΑΣ #10:

```
function calcDistance(x1,y1,x2,y2)
{
	return Math.sqrt(Math.pow((x2 - x1),2) + Math.pow((y2 - y1),2)) ;
}
```

Παρατηρούμε ότι δοθέντων δύο σημείων x1,y1 και x2,y2 στην ουσία η συνάρτηση υλοποιεί τον μαθηματικό τύπο που δίνει την απόσταση μεταξύ των δύο σημείων και είναι ο:

$$d = \sqrt{(x2 - x1)^2 + (y2 - y1)^2}$$

Επίσης, πρέπει να υπολογιστεί και η γωνία του ευθύγραμμου τμήματος μεταξύ των δύο σημείων, ώστε να περιστραφεί κατάλληλα το πολύγωνο του τοίχου για να προσανατολιστεί σωστά. Η γωνία του ευθύγραμμου τμήματος μεταξύ δύο σημείων υπολογίζεται από:

$$\theta = \operatorname{atan}\left(\frac{y2 - y1}{x2 - x1}\right)$$

Η συνάρτηση που υπολογίζει την γωνία μεταξύ δύο σημείων περιέχεται στο αρχείο utilities.js και ονομάζεται calcAngle(). Ο κώδικάς της παρουσιάζεται παρακάτω:

ΚΩΔΙΚΑΣ #11:

```
1.  function calcAngle(x1,y1,x2,y2)
2.  {
3.  if (x2 == x1 && y2 == y1)  return 0;

4.  var angle = (Math.atan((y2 - y1) / (x2 - x1))) * 180 / Math.PI;

5.  if (x2 >= x1 && y2 < y1) {
6.  angle = angle - 180;
7.  }

8.  if (x2 < x1 || y2 < y1) {
9.  angle = angle + 180;
10. }

11. return angle;
12. }
```

Επίσης όσον αφορά της γωνίες έχουν υλοποιηθεί και δύο βοηθητικές συναρτήσεις οι οποίες εκτελούν την μετατροπή μονάδων γωνίας από ακτίνια σε μοίρες και το αντίστροφο. Πρόκειται για τις συναρτήσεις degToRad() και radToDeg():

ΚΩΔΙΚΑΣ #12:

```
1. function degToRad(a)
2. {
3. return a* (Math.PI / 180);
4. }

5. function radToDeg(a)
6. {
7. return a * (180 / Math.PI);
8. }
```

Στην createWallShape(), με την κλήση των συναρτήσεων calcDistance() και calcAngle() υπολογίζεται το μήκος και ν προσανατολισμός του τοίχου πάντα με σημείο αναφοράς τις συντεταγμένες του Handle έναρξης. Επίσης καλείται η συνάρτηση createElementNS() για να δημιουργηθεί ένα SVG αντικείμενο τύπου <polygon>. Κάθε SVG αντικείμενο τύπου <polygon> περιέχει ένα array με τα σημεία που το καθορίζουν εν ονόματι points. Χρησιμοποιείται η μέθοδος points.append() στο πολύγωνο, για να προστεθούν 6 νέα σημεία, οι συντεταγμένες των οποίων, καθορίζονται πλήρως από το μήκος και το πάχος του τοίχου. Κατόπιν δημιουργείται ένα νέο στοιχείο SVG τύπου <g> για να δράσει σαν container, στο οποίο θα γίνει append το SVG element <polygon>. Δημιουργείται και ένα δεύτερο SVG element <g>, το οποίο αποτελεί το εξωτερικό στοιχείο που περιέχει όλα τα γραφικά που αφορούν το συγκεκριμένο αντικείμενο τύπου Wall. Σε αυτό το <g> στοιχείο, εφαρμόζεται κατάλληλος γεωμετρικός μετασχηματισμός, για να μετατοπιστεί το πολύγωνο του τοίχου και να περιστραφεί, έτσι ώστε να είναι ευθυγραμμισμένο με το ευθύγραμμο τμήμα που ενώνει τα δύο handles. Καταρχήν πραγματοποιείται μια μετατόπιση κατά x1,y1 (όπου x1,y1 οι συντεταγμένες του start Handle). Έτσι το σημείο του πολυγώνου με index=0 το οποίο βρισκότανε στην αρχή των αξόνων τώρα ταυτίζεται με το σημείο του αντικειμένου start Handle. Επίσης εφαρμόζεται και μια περιστροφή κατά γωνία angle την οποία υπολόγισε η calcAngle() και το πολύγωνο περιστρέφεται ώστε το σημείο του με index=2 να συμπέσει με το σημείο του end Handle. Αναφέρθηκε σε προηγούμενη παράγραφο η διαδικασία ομαλής σύνδεσης τον τοίχων σε έναν κόμβο. Η διαδικασία αυτή υλοποιείται με την μέθοδο calibrateWalls() η οποία αποτελεί μέλος του αντικειμένου τύπου Handle.

Ο κώδικας της μεθόδου παρουσιάζεται παρακάτω:

ΚΩΔΙΚΑΣ #13:

```
1. // Calibrate wall joints
2. ////////////////////////////
3. function WallEl(w,a,f)
4. {
5. this.w = w;
6. this.angle = a;
7. this.flip = f;
8. }
9. function sortWalls(a,b)
10. {
11. return (a.angle) - (b.angle);
12. }
```

```
13. Handle.prototype.calibrateWalls = function()
14. {
15. if (this.walls.length < 2)  return;
16. this.fillArray = new Array();
17. warray = new Array();
18. var I;
19. var ang;
20. var flp;
21. var wel;
22. for (I in this.walls)
23. {
24. ang = this.walls[i].rightAngle(this);
25. if (this.walls[i].checkHandle(this) == 1)
26. {
27. flp = true;
28. }
29. else
30. {
31. flp = false;
32. }
33. wel = new WallEl(this.walls[i],ang,flp);
34. warray.push(wel);
35. }
36. warray.sort(sortWalls);
37. for (I in warray) {
38. if (I < warray.length - 1) {
39. var w1 = warray[i];
40. var w2 = warray[Number(i) + 1];
41. }
42. else {
43. var w1 = warray[i];
44. var w2 = warray[0];
45. }
46. // the four crucial points
47. //the four transformation matrices
48. var mx1 = w1.w.SVG.getCTM();
49. var mx2 = w2.w.SVG.getCTM();
50. var imx1 = mx1.inverse();
51. var imx2 = mx2.inverse();
52. var poly1 = w1.w.poly;
53. var poly2 = w2.w.poly;
54. var pr1;
55. var pr2;
56. var p1 = SVGroot.createSVGPoint();
57. var p2 = SVGroot.createSVGPoint();
58. var p3 = SVGroot.createSVGPoint();
59. var p4 = SVGroot.createSVGPoint();
60. var d1 = w1.w.depth / 2 ;
61. var d2 = w2.w.depth / 2 ;
62. var wd1 = w1.w.getWidth();
63. var wd2 = w2.w.getWidth();
64. if (w1.flip == false) {
```

```
65. p1.x = 0;
66. p1.y = -d1;
67. p2.x = wd1;
68. p2.y = -d1;
69. pr1 = 2;
70. }
71. else {
72. p1.x = wd1;
73. p1.y = d1;
74. p2.x = 0;
75. p2.y = d1;

76. pr1 = 5;
77. }
78. if (w2.flip == false) {
79. p3.x = 0;
80. p3.y = d2;
81. p4.x = wd2;
82. p4.y = d2;
83. pr2 = 4;
84. }
85. else {
86. p3.x = wd2;
87. p3.y = -d2;
88. p4.x = 0;
89. p4.y = -d2;
90. pr2 = 1;
91. }
92. p1 = p1.matrixTransform(mx1);
93. p2 = p2.matrixTransform(mx1);
94. p3 = p3.matrixTransform(mx2);
95. p4 = p4.matrixTransform(mx2);
96. var pi = lineIntersect(p1, p2, p3, p4);
97. if (pi == null) {
98. //poly1.points.getItem(pr1).x = -10;
99. //poly1.points.getItem(pr1).y = pi1.y;
100.       //
101.       //poly2.points.getItem(pr2).y = pi2.y;
102.       }else
103.       {
104.       var pi1 = pi.matrixTransform(imx1);
105.       var pi2 = pi.matrixTransform(imx2);

106.       poly1.points.getItem(pr1).x = pi1.x;

107.       //poly1.points.getItem(pr1).y = pi1.y;

108.       poly2.points.getItem(pr2).x = pi2.x;

109.       //poly2.points.getItem(pr2).y = pi2.y;
110.       }
111.       w1.w.updateSides();
112.       w2.w.updateSides()
113.       }
114.       this.fillArray = new Array();

115.       for (I in warray)

116.       {
117.       parray = warray[i].w.poly.points;

118.       matrix = warray[i].w.SVG.getCTM();

119.       var p3 = SVGroot.createSVGPoint();

120.       if (warray[i].flip == false) {
```

```
121.      var p1 = parray.getItem(2);
122.      var p2 = parray.getItem(4);

123.      if (p1.x < p2.x)

124.      {

125.      p3.x = p1.x;

126.      p3.y = p2.y;
127.      }
128.      else
129.      {
130.      p3.x = p2.x;
131.      p3.y = p1.y;
132.      }
133.      }
134.      else
135.      {
136.      var p1 = parray.getItem(5);
137.      var p2 = parray.getItem(1);

138.      if (p1.x > p2.x)

139.      {

140.      p3.x = p1.x;

141.      p3.y = p2.y;
142.      }
143.      else
144.      {
145.      p3.x = p2.x;
146.      p3.y = p1.y;
147.      }
148.      }
149.      p2 = p2.matrixTransform(matrix);
150.      p3 = p3.matrixTransform(matrix);

151.      this.fillArray.push(p2);

152.      this.fillArray.push(p3);
153.      }
154.}
```

Καταρχήν χρησιμοποιείται η βοηθητική δομή WallEl η οποία περιέχει τρία attributes:

- **W:** αναφορά προς ένα συγκεκριμένο wall object.
- **Angle:** η γωνία προσανατολισμού του συγκεκριμένου wall object.
- **Flip:** αν το συγκεκριμένο handle στο οποίο συνδέεται ο τοίχος είναι handle έναρξης τότε ο τοίχος είναι «ανεστραμμένος» αλλιώς αν είναι handle τέλους ο τοίχος είναι «μη-ανεστραμμένος».

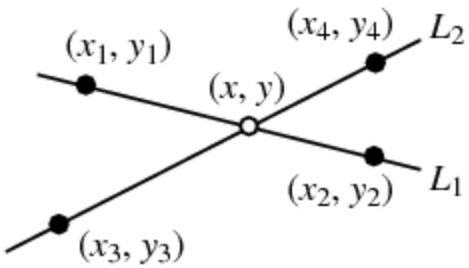
Παρατηρούμε ότι υπάρχει και η συνάρτηση sort-walls η οποία εξετάζει τις γωνίες των τοίχων και τους βάζει στην σειρά σύμφωνα με τον κανόνα του ρολογιού.

Καλείται η calibrateWalls() η οποία εξετάζει το array walls του συγκεκριμένου Handle object. Αν βρεθούν λιγότερα από δύο στοιχεία η calibrateWalls() επιστέφει

γιατί υποτίθεται ότι δεν υπάρχει σύνδεση μεταξύ δύο ή παραπάνω τοίχων στο συγκεκριμένο Handle. Διαφορετικά συνεχίζεται η εκτέλεση και ακολουθεί βρόγχος (κώδικας #13, γραμμή #12) ο οποίος εξετάζει ένα-ένα τα στοιχεία του array Walls στο συγκεκριμένο handle object και δημιουργεί ένα νέο array αντικειμένων WallEl, στα οποία αποθηκεύει αναφορά σε κάθε τοίχο, την γωνία του και το αν είναι ανεστραμμένος η όχι. Αφού δημιουργηθεί η διάταξη warray των αντικειμένων WallEl, εκτελείται διαδικασία ταξινόμησής τους με βάση τις γωνίες προσανατολισμού τους από την μικρότερη στην μεγαλύτερη. Αυτό είναι πολύ σημαντικό ώστε να ταξινομηθούν οι τοίχοι με βάση την φορά του ρολογιού διότι έτσι διαδικασία θα τους «επισκεφτεί» κατά ζευγάρια. Αυτό κάνει ο επόμενος βρόγχος ο οποίος ξεκινάει από το πρώτο στοιχείο του warray και διαλέγει αυτό και το αμέσως επόμενο αποθηκεύοντας τις αναφορές τους στις μεταβλητές w1 και w2. Όταν κατά την εκτέλεση του βρόγχου η διαδικασία φτάσει στο τελευταίο στοιχείο του warray, το επιλέγει σαν w1 και σαν w2 επιλέγει το πρώτο και ύστερα σπάει ο βρόγχος.

Αυτό που γίνεται κατόπιν στο εσωτερικό του συγκεκριμένου βρόγχου είναι ότι δημιουργούνται 4 σημεία SVG points με ονόματα p1,p2,p3,p4. Αυτά τα σημεία πρόκειται να περιγράψουν τις δύο ακμές των πολυγώνων, οι οποίες θα εξεταστούν για το πού τέμνονται. Για να εντοπιστούν οι ευθείες, πρέπει να ελεγχθεί πρώτα αν ο κάθε τοίχος είναι ανεστραμμένος η όχι, για να γίνει γνωστό πια πλευρά του κάθε πολυγώνου θα επιλεγεί. Αφού γίνει η επιλογή των κατάλληλων πλευρών των πολυγώνων, πρέπει να εφαρμοστούν γραμμικοί μετασχηματισμοί στα σημεία αυτών, ώστε να γίνει μεταφορά στο αρχικό σύστημα αξόνων. Αυτό επιτυγχάνεται παίρνοντας τους πίνακες CTM για τα πολύγωνα του πρώτου και του δεύτερου τοίχου και εφαρμόζοντας τον μετασχηματισμό που περιγράφουν στα σημεία των πολυγώνων. Κατόπιν αφού τα σημεία μεταφερθούν στις συντεταγμένες της περιοχής σχεδίασης, μπορούμε να χρησιμοποιηθούν για να υπολογιστεί το σημείο τομής των δύο ευθειών που περιγράφουν. Αυτό γίνεται με την κλήση της συνάρτησης lineIntersect() η οποία δέχεται 4 σημεία που περιγράφουν 2 ευθείες και υπολογίζει το σημείο τομής των ευθειών το οποίο αποθηκεύεται στην μεταβλητή pi. Η lineInstersect() υλοποιεί τον εξής μαθηματικό τύπο για την τομή δύο ευθύγραμμων τμημάτων :

Έστω δύο ευθύγραμμα τμήματα L1 και L2:



Η τομή τους δίνεται από την λύση του συστήματος:

$$\begin{vmatrix} x & y & 1 \\ x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \end{vmatrix} = 0$$

$$\begin{vmatrix} x & y & 1 \\ x_3 & y_3 & 1 \\ x_4 & y_4 & 1 \end{vmatrix} = 0$$

Η οποία δίνεται για τα x και y από τους τύπους:

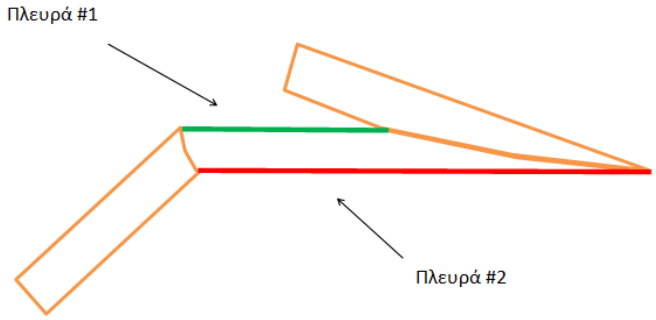
$$x = \frac{\begin{vmatrix} \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} & \begin{vmatrix} x_1 & 1 \\ x_2 & 1 \end{vmatrix} \\ \begin{vmatrix} x_3 & y_3 \\ x_4 & y_4 \end{vmatrix} & \begin{vmatrix} x_3 & 1 \\ x_4 & 1 \end{vmatrix} \end{vmatrix}}{\begin{vmatrix} \begin{vmatrix} x_1 & 1 \\ x_2 & 1 \end{vmatrix} & \begin{vmatrix} y_1 & 1 \\ y_2 & 1 \end{vmatrix} \\ \begin{vmatrix} x_3 & 1 \\ x_4 & 1 \end{vmatrix} & \begin{vmatrix} y_3 & 1 \\ y_4 & 1 \end{vmatrix} \end{vmatrix}} = \frac{\begin{vmatrix} \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} & x_1 - x_2 \\ \begin{vmatrix} x_3 & y_3 \\ x_4 & y_4 \end{vmatrix} & x_3 - x_4 \end{vmatrix}}{\begin{vmatrix} x_1 - x_2 & y_1 - y_2 \\ x_3 - x_4 & y_3 - y_4 \end{vmatrix}}$$

$$y = \frac{\begin{vmatrix} \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} & \begin{vmatrix} y_1 & 1 \\ y_2 & 1 \end{vmatrix} \\ \begin{vmatrix} x_3 & y_3 \\ x_4 & y_4 \end{vmatrix} & \begin{vmatrix} y_3 & 1 \\ y_4 & 1 \end{vmatrix} \end{vmatrix}}{\begin{vmatrix} \begin{vmatrix} x_1 & 1 \\ x_2 & 1 \end{vmatrix} & \begin{vmatrix} y_1 & 1 \\ y_2 & 1 \end{vmatrix} \\ \begin{vmatrix} x_3 & 1 \\ x_4 & 1 \end{vmatrix} & \begin{vmatrix} y_3 & 1 \\ y_4 & 1 \end{vmatrix} \end{vmatrix}} = \frac{\begin{vmatrix} \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} & y_1 - y_2 \\ \begin{vmatrix} x_3 & y_3 \\ x_4 & y_4 \end{vmatrix} & y_3 - y_4 \end{vmatrix}}{\begin{vmatrix} x_1 - x_2 & y_1 - y_2 \\ x_3 - x_4 & y_3 - y_4 \end{vmatrix}},$$

Το σημείο τομής πρέπει να μετασχηματιστεί στο σύστημα συντεταγμένων του πολυγώνου κάθε τοίχου, ώστε να υπάρχει αναφορά για το που θα γίνει η μετακίνηση σημείων σύμφωνα με την διαδικασία. Η μεταφορά του σημείου τομής από το σύστημα συντεταγμένων της περιοχής σχεδίασης στο σύστημα συντεταγμένων του εκάστοτε πολυγώνου γίνεται με γραμμικό μετασχηματισμό χρησιμοποιώντας τον αντίστροφο του πίνακα CTM κάθε πολυγώνου. Ταυτόχρονα με την εκτέλεση της calibrateWalls() συλλέγονται με την σειρά ένα-ένα τα σημεία τομής των πλευρών των πολυγώνων και αποθηκεύονται στο array με την ονομασία fillArray το οποίο χρησιμοποιείται αργότερα στην δημιουργία της τρισδιάστατης γεωμετρίας.

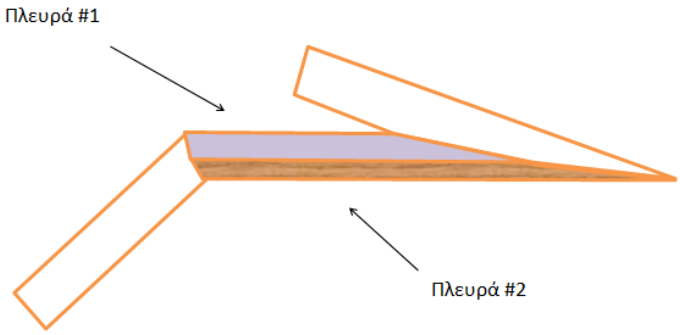
5.5.7 Υλοποίηση διακόσμησης τοίχων

Κάθε τοίχος αποτελείται από δύο (2) πλευρές οι οποίες ανήκουν σε διαφορετικά δωμάτια. Το κατά πώς είναι προσανατολισμένος ο τοίχος και το κατά πώς συνδέεται με τους άλλους τοίχους επηρεάζουν το μήκος των πλευρών αυτών. Οι πλευρές αυτές επισημαίνονται στο παρακάτω σχήμα:



Εικόνα 93 – Οι δύο πλευρές ενός τοίχου.

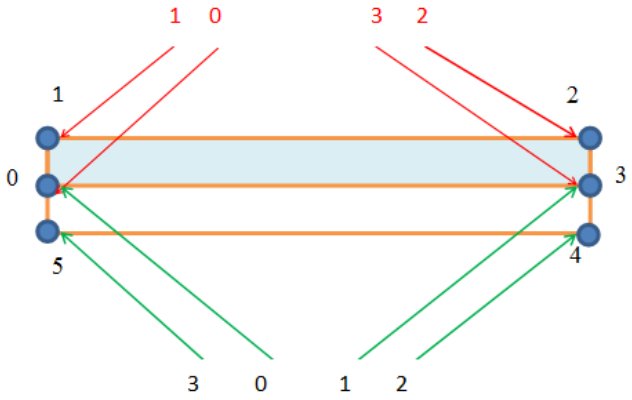
Για να παρέχεται καλύτερη εποπτεία στον χρήστη για το ποια υφή ή χρώμα είναι εφαρμοσμένο στην πλευρά κάθε τοίχου δημιουργήθηκαν δύο ακόμα πολύγωνα πάνω από το πολύγωνο του τοίχου τα οποία το μοιράζουν στην μέση. Τα πολύγωνα αυτά παρατηρούνται στο παρακάτω σχήμα:



Εικόνα 94 – Επισκόπηση της διακόσμησης δύο πλευρών ενός τοίχου.

Έτσι δίνεται η δυνατότητα να εφαρμοστεί ξεχωριστό γέμισμα (fill) στο καθένα πολύγωνο, το οποίο να αντιπροσωπεύει την υφή ή το χρώμα το οποίο έχει εφαρμοστεί στην συγκεκριμένη πλευρά του τοίχου. Στο παραπάνω σχήμα η πλευρά #1 του τοίχου φαίνεται ότι χρησιμοποιεί μωβ χρώμα για την διακόσμηση της ενώ στην πλευρά #2 υπάρχει υφή τύπου ξύλου.

Τα δύο αυτά πολύγωνα δημιουργούνται καλώντας την μέθοδο updateWallSides() του αντικειμένου Wall και δημιουργούν δύο attributes στο τρέχων αντικείμενο με τις ονομασίες polyA και polyB. Οι attributes αυτές περιέχουν αναφορές στα 2 SVG element <polygon> τα οποία αποτελούν την γραφική απεικόνιση των πλευρών του τοίχου. Στην παρακάτω εικόνα παρουσιάζουμε την αντιστοίχηση των σημείων των 2 νέων πολυγώνων σε σχέση με τα σημεία του αρχικού πολυγώνου που αναπαριστά την ολότητα του τοίχου:



Εικόνα 95 – Τα δύο πολύγωνα που αναπαριστούν την διακόσμηση ενός τοίχου.

5.3.8 Υλοποίηση λογικής οριοθέτησης δωματίων

Όπως αναφέρθηκε και στο προηγούμενο κεφάλαιο αφού σχεδιαστούν οι τοίχοι, οι οποίοι αποτελούν τον «σκελετό» του σχεδίου κατόψεως, μπορούν να οριοθετηθούν τα δωμάτια. Για την οριοθέτησή τους, χρησιμοποιούνται ως σημεία αναφοράς τα αντικείμενα τύπου Handle, τα οποία στην ουσία αποτελούν τα σημεία των γωνιών ενός δωματίου. Η υλοποίηση της οριοθέτησης των δωματίων περιλαμβάνεται στο αρχείο floor.js. Εκεί δηλώνεται η κλάση αντικειμένων Floor η οποία αναλαμβάνει να ρυθμίσει της παραμέτρους της γραφικής αναπαράστασης των δαπέδων των δωματίων αλλά και των οροφών. Η κλάση Floor δηλώνεται στον κώδικα ως εξής:

ΚΩΔΙΚΑΣ #14:

```
1.  function Floor()
2.  {
3.  this.id = Floors.getNextId();
4.  this.points = new Array();
5.  this.SVG = null;
6.  this.name = null;
7.  this.type = null;

8.  this.color = new Color(155,155,155);
9.  this.texture = null;
10. this.colorC = new Color(155,155,155);
11. this.textureC = null;

12. this.typeF = "paint";
13. this.typeC = "paint";


14. Floors.addFloor(this);

15. }
```

Η κλάση μοιράζεται παρόμοιες attributes με τις κλάσεις Handle και Wall όπως id, SVG(για την αναφορά των γραφικών elements SVG) κτλ. Επιπρόσθετα περιλαμβάνει attributes όπως name (το αλφαριθμητικό με την ονομασία του δωματίου) και type (τύπος του δωματίου). Βασική ιδιότητα αποτελεί η points η οποία στην ουσία είναι μια λίστα των Handles που οριοθετούν το δωμάτιο. Επίσης περιλαμβάνει την ιδιότητα texture η οποία περιέχει αναφορά στην υφή που διακοσμεί το δάπεδο, την ιδιότητα color η οποία περιέχει αναφορά στο χρώμα που διακοσμεί το δάπεδο και αντίστοιχα της ιδιότητες colorC και textureC για την οροφή. Τέλος περιλαμβάνει και τις ιδιότητες typeF και typeC οι οποίες καθορίζουν τον τύπο διακόσμησης του δαπέδου και της οροφής αντίστοιχα. Η συνάρτηση createFloorShape() έχει αντίστοιχη λειτουργικότητα με τις συναρτήσεις createHandleShape() και createWallShape(). Δημιουργεί τα απαραίτητα SVG Elements τα οποία αναπαριστούν το δάπεδο του δωματίου. Μια άλλη σημαντική παράμετρος η οποία έχει ιδιαίτερη χρησιμότητα στην κατασκευή της τρισδιάστατης γεωμετρίας είναι το αν η δήλωση των σημείων που οριοθετούν το πολύγωνο του δωματίου έχει γίνει με την φορά των δεικτών του ρολογιού ή αντίθετα. Αν είναι αντίθετα δηλαδή counter clock wise ή ccw τότε αυτό ελέγχεται από την μέθοδο Floor.isCCW() Ο κώδικας της μεθόδου παρουσιάζεται παρακάτω και χρησιμοποιεί τον αλγόριθμο Graham Scann για να υπολογίσει την φορά δήλωσης των σημείων του πολυγώνου:

ΚΩΔΙΚΑΣ #15:

```
1.  Floor.prototype.isCCW = function()
2.  {
3.  if (this.points.length < 0) return;
4.  var r;
5.  var mul=1;
6.  var i;
7.  for (i=0;i<this.points.length-2;i++)
8.  {
9.  r = ccw(this.points[i],this.points[i+1],this.points[i+2]);
10. if (r==0) r = 1;
11. mul = mul * r;
12. }
13. //console.log(mul);
14. if (mul < 0) return true;
15. else return false;
16. }
17. function ccw(p1, p2, p3){
18. return (p2.x - p1.x) * (p3.y - p1.y) - (p2.y - p1.y) *     (p3.x -
    p1.x)
19. }
```

Η μέθοδος περιλαμβάνει βρόγχο, ο οποίος εξετάζει ένα προς ένα τα σημεία του array points και καλεί για καθένα από αυτά την συνάρτηση ccw() διοχετεύοντάς τις συντεταγμένες του σημείου σαν πρώτο όρισμα και ακολουθούν οι συντεταγμένες του επόμενου και του παρά-επόμενου σημείου στο array points. Η ccw(), σαν συνάρτηση δέχεται ως ορίσματα τρία σημεία, τα οποία ελέγχει, αν στην σειρά που βρίσκονται

δημιουργούν πορεία αντίθετη με τους δείκτες του ρολογιού ή όχι. Αυτό δίνεται από τον τύπο στην γραμμή 18. Επίσης στο ίδιο αρχείο κώδικα περιλαμβάνεται και μια κλάση FloorArray, η οποία όπως και με τις HandleArray και WallArray, περιλαμβάνει μια λίστα με αναφορές στα αντικείμενα της κλάσης Floor που έχουν ήδη δημιουργηθεί.

5.3.9 Υλοποίηση λογικής των αντικειμένων που αναπαριστούν πόρτες, παράθυρα και οπές

Η υλοποίηση των πορτών ή των παραθύρων στους τοίχους, γίνεται με τις κλάσεις Windoor και WindoorArray οι οποίες περιγράφονται στο αρχείο Javascript με την ονομασία windoor.js. Η κλάση Windoor έχει παρόμοια δομή με τις κλάσεις που περιγράφηκαν παραπάνω (Handle, Wall). Παρομοίως η κλάση WindoorArray αποτελεί παρόμοια υλοποίηση των κλάσεων HandleArray,WallArray αλλά για αντικείμενα τύπου Windoor.

ΚΩΔΙΚΑΣ #16:

```
1.  function Windoor(x,y,type,w,h,e)
2.  {
3.  this.id = Windoors.getNextId();
4.  this.x = x;
5.  this.y = y;

6.  this.SVG = null;
7.  this.backSVG = null;
8.  this.content = null;
9.  this.side = null;

10. this.wall=null;
11. this.type = type;

12. this.wallx=0;

13. this.flip = 0;

14. this.depth = 20;

15. this.width = Number(w);
16. this.height = Number(h);
17. this.elevate = Number(e);

18. this.asset = null;

19. Windoors.addWindoor(this);
20. this.draw();
21. }
```

Σημαντικές μέθοδοι της κλάσης Windoor είναι οι attach() και detach(). Η attach() καλείται περίπτωση που χρειάζεται ένα αντικείμενο της κλάσης (πόρτα, παράθυρο) να τοποθετηθεί πάνω στον τοίχο. Στόχος της είναι η τοποθέτηση του γραφικού στοιχείου που αναπαριστά το παράθυρο/πόρτα μέσα στο στοιχείο <g> του τοίχου με

κατάλληλη αλλαγή του συστήματος συντεταγμένων. Έτσι το παράθυρο/πόρτα εγκαταλείπει το αρχικό σύστημα συντεταγμένων και υιοθετεί αυτό του αντίστοιχου τοίχου. Αυτό έχει σαν αποτέλεσμα τον αυτόματο προσανατολισμό του αντικειμένου πάνω στον τοίχο. Η μέθοδος detach() αναλαμβάνει την επαναφορά του αντικειμένου windoor στο αρχικό σύστημα συντεταγμένων της περιοχής σχεδίασης.

5.3.10 Υλοποίηση δομών αντικειμένων διακόσμησης

Τα διάφορα αντικείμενα που φορτώνονται από βάση δεδομένων και τοποθετούνται μέσα στο σχέδιο κατόψεως, περιγράφονται από την κλάση αντικειμένων deco η οποία δηλώνεται στο αρχείο deco.js. Ο κώδικας της κλάσης είναι ο εξής:

ΚΩΔΙΚΑΣ #17:

```
1. function Deco(x,y,type,w,h,e)
2. {
3. this.id =Decos.getNextId();
4. this.x = x;
5. this.y = y;

6. this.TSVG = null;
7. this.RSVG = null;
8. this.content = null;

9. this.asset = null;

10. Decos.addDeco(this);
11. this.draw();
12. }
```

Παρατηρούμε ότι και αυτή η κλάση αντικειμένων περιλαμβάνει παρόμοια attributes με τις άλλες κλάσεις, που περιγράφουν αντικείμενα του τύπου Handle, Wall, Floor, Windoor κτλ. Περιλαμβάνει τα attributes x,y για τις συντεταγμένες του κεντρικού σημείου του αντικειμένου, την attribute id, επίσης τρείς attributes που λαμβάνουν αναφορές σε SVG στοιχεία:

- **Content** = περιλαμβάνει αναφορά στα γραφικά στοιχεία SVG τα οποία αναπαριστούν το αντικείμενο και φορτώνονται από την βάση δεδομένων.
- **RSVG** = περιλαμβάνει αναφορά στο SVG στοιχείο <g> το οποίο περικλείει το περιεχόμενο των γραφικών και λαμβάνει μετασχηματισμό περιστροφής ώστε να δίνεται η δυνατότητα να ελέγχουμε την περιστροφή του αντικειμένου deco γραφικά.
- **TSVG** = περιλαμβάνει αναφορά στο στοιχείο SVG το οποίο δρα ως container του SVG element <g> που περιγράφηκε προηγουμένως (υπεύθυνο για την περιστροφή). Το νέο αυτό <g> λαμβάνει μετασχηματισμό μετατοπίσεως ώστε να ελέγχουμε την μετατόπιση του αντικειμένου deco γραφικά.

Η attribute asset περιλαμβάνει αναφορά στην δομή που περιγράφει το asset για το συγκεκριμένο τύπο παραθύρου. Στις μεθόδους περιλαμβάνονται δύο απλές συναρτήσεις οι οποίες είναι υπεύθυνες για την περιστροφή και την μετατόπιση του στοιχείου και είναι οι deco.move(x,y) και οι deco.rotate(a). Η πρώτη, δέχεται σαν ορίσματα τις συντεταγμένες ενός σημείου του επιπέδου και μετακινεί εκεί το αντικείμενο deco. Η δεύτερη, δέχεται σαν όρισμα μία γωνία και περιστρέφει ανάλογα το αντικείμενο deco. Ο κώδικας των δύο συναρτήσεων περιγράφεται παρακάτω:

ΚΩΔΙΚΑΣ #18:

```
1.  Deco.prototype.move = function(x,y)
2.  {
3.  this.x=x;
4.  this.y=y;

5.  this.TSVG.setAttributeNS(null,"transform","translate(" + x + "," + y +
    ")");

6.  }

7.  Deco.prototype.rotate = function(a)
8.  {
9.  this.angle = a;

10. this.RSVG.setAttributeNS(null,"transform","rotate(" + a + ")");

11. }
```

5.4 Ασύγχρονη φόρτωση αντικειμένων από την βάση δεδομένων στην μνήμη του φυλλομετρητή (τεχνολογία AJAX)

Κατά την διάρκεια τις σχεδίασης, πολλά από τα αντικείμενα που επιλέγει ο χρήστης να προσθέσει στην κάτοψη, είναι καταχωρημένα στην βάση δεδομένων της εφαρμογής η οποία λειτουργεί στην πλευρά του διακομιστή. Είναι αναγκαίο λοιπόν να διαβαστούν από τον διακομιστή ιστού και να μεταφερθούν σε πλευρά client. Η επικοινωνία αυτή, γίνεται ασύγχρονα, με αποτέλεσμα να μην χρειάζεται φόρτωση μιας νέας ή ανανέωση της ιδίας σελίδας και έτσι ο χρήστης βιώνει μια συνεχή ροή στην αλληλεπίδραση, αντίστοιχη μιας desktop εφαρμογής. Όπως περιγράψαμε και στα προηγούμενα κεφάλαια, η τεχνολογία που το επιτρέπει αυτό ονομάζεται ajax (Asynchronous Javascript And XML) και απαιτεί χρησιμοποίηση ενός ειδικού αντικειμένου σε Javascript (XMLHttpRequest). Το αντικείμενο αυτό δίδει και ζητάει δεδομένα από ένα script στην πλευρά του server, το οποίο είναι γραμμένο σε μια από τις αντίστοιχες server-side γλώσσες προγραμματισμού (PHP, asp κτλ). Στην συγκεκριμένη εφαρμογή και διατριβή, η γλώσσα που εκτελείται και υλοποιεί την λογική της εφαρμογής στην πλευρά του διακομιστή είναι η PHP. Η τεχνική του Ajax λοιπόν απαιτεί συνεργασία Javascript και PHP για την επιτυχή επικοινωνία του

φυλλομετρητή (πελάτης) και του διακομιστή. Υπάρχουν αντικείμενα τα οποία περιγράφονται από κλάσεις που αναφέρθηκαν σε προηγούμενες παραγράφους τα οποία περιλαμβάνουν τύπους και παραμέτρους που είναι αποθηκευμένα στην βάση δεδομένων. Π.χ. για τα αντικείμενα τύπου windoor (δηλαδή αντικείμενα που περιγράφουν παράθυρα και πόρτες) υπάρχει ειδικός πίνακας ο οποίος περιέχει δεδομένα για διάφορους τύπους παραθύρων και πορτών. Το script για την κατασκευή του συγκεκριμένου πίνακα στην βάση δεδομένων ήταν το εξής:

ΚΩΔΙΚΑΣ #19:

```
CREATE TABLE `windoors` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `title` text NOT NULL,
  `description` text NOT NULL,
  `type` varchar(45) NOT NULL,
  `svg` text NOT NULL,
  `x3d` text NOT NULL,
  `width` int(10) unsigned NOT NULL,
  `height` int(10) unsigned NOT NULL,
  `elevate` int(10) unsigned NOT NULL,
  `cost` float NOT NULL,
  `img` text NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=latin1;
```

Παρατηρούμε δηλαδή ότι ο πίνακας περιλαμβάνει τις εξής στήλες:

- **Id:** το διαχωριστικό κλειδί της κάθε εγγραφής.
- **Title:** η ονομασία του κάθε προϊόντος (παράθυρο / πόρτα).
- **Description:** μια σύντομη περιγραφή.
- **Type:** ο τύπος (πόρτα/παράθυρο).
- **SVG:** αναφορά στο αρχείο SVG που περιέχει την δισδιάστατη γραφική αναπαράσταση του αντικειμένου.
- **X3D:** αναφορά στο αρχείο X3D που περιέχει την τρισδιάστατη γεωμετρική αναπαράσταση του αντικειμένου.
- **Width:** το μήκος της πόρτας/παράθυρου.
- **Height:** το ύψος της πόρτας/παράθυρου.
- **Elevate:** το ύψος από το έδαφος αν πρόκειται για παράθυρο
- **Cost:** το κόστος.
- **Img:** η αντιπροσωπευτική εικόνα μικρογραφία που εμφανίζεται στην παλέτα επιλογής του αντικειμένου μέσα στο gui της εφαρμογής (συνήθως στην πλευρική στήλη του δισδιάστατου editor σχεδίασης κατόψεων).

Επειδή η πρόσβαση στον server κοστίζει σε χρόνο, κάθε φορά που ο χρήστης επιλέγει να προσθέσει ένα συγκεκριμένο παράθυρο στο σχέδιο, θα πρέπει να περιμένει να μεταφερθούν όλες οι παράμετροι που το περιγράφουν από την βάση, κάτι το οποίο δυσχεραίνει την ταχύτητα της εφαρμογής. Για αυτό τον λόγο λοιπόν, επιλέξαμε να δημιουργήσουμε δομές δεδομένων, οι οποίες χρησιμοποιούνται σαν αποθήκες των ξεχωριστών τύπων αντικειμένων που διαβάζονται από την βάση. Έτσι π.χ. διατηρείται στην μνήμη του φυλλομετρητή μια λίστα αντικειμένων windoor Assets η οποία αποθηκεύει τα δεδομένα όλων των ξεχωριστών τύπων αντικειμένων τύπου Windoor, που έχουν ζητηθεί από την βάση μέχρι εκείνη την στιγμή. Αν ο χρήστης επιλέξει να προσθέσει στο σχέδιο ένα παράθυρο το οποίο έχει ήδη ξανά προστεθεί (δηλ έχει ήδη ζητηθεί από τον server), τότε το πρόγραμμα θα αναζητήσει της παραμέτρους στο κατάλληλο windoor Asset που έχει δημιουργηθεί και όχι στην βάση, επιταχύνοντας έτσι την διαδικασία εισαγωγής. Τα αντικείμενα σχετικά με τα windoor Assets περιγράφονται από την κλάση αντικειμένων wdAsset η οποία βρίσκεται στο αρχείο wdAsset.js.

Η δήλωση της κλάσης είναι η εξής:

ΚΩΔΙΚΑΣ #20:

```
function WdAsset()
{
	this.id =null;
	this.img = null;
	this.def =null;
	this.defurl = null;
	this.w = null;
	this.h = null;
	this.e = null;
	this.title = null;
	this.desc = null;
	this.cost = null;
	this.SVG = null;
	this.X3D = null;
	this.type = null;
	//

}
```

Παρατηρούμε ότι τα properties της κλάσης συμπίπτουν σχεδόν με τις στήλες του πίνακα “windoors” της βάσης δεδομένων. Επειδή τα αντικείμενα windoors αναπαριστώνται από SVG elements στην περιοχή σχεδίασης, τα οποία επαναχρησιμοποιούνται στην σκηνή, θεωρήθηκε σωστό να καταχωρούνται σαν αντικείμενα <def> στον κώδικα SVG. Τα αντικείμενα αυτά δηλώνονται μια φορά αλλά μπορούν να χρησιμοποιηθούν πολλές, με την χρήση των στοιχείων <use>. Για αυτό τον λόγο, υπάρχουν τα properties def και defurl σαν αναφορές στο στοιχείο <def> που θα δημιουργηθεί για το συγκεκριμένο asset και στο αντίστοιχο defurl που

θα επιτρέπει την επαναχρησιμοποίηση του αρκετές φορές μέσα στην περιοχή σχεδίασης. Στο αρχείο wdAssets.js δηλώνεται επίσης η κλάση αντικειμένων wdArray η οποία στην ουσία αποτελεί μια λίστα αντικειμένων τύπου wdAssets και έχει μόνο μία property: την wdArray.list. Η συγκεκριμένη property αποτελεί array αναφορών σε αντικείμενα wdAssets. Η μέθοδος wdArray.add() προσθέτει μια αναφορά στο array wdArray.list για ένα νέο αντικείμενο που δημιουργήθηκε, ενώ η wdArray.remove() αντίστοιχα αφαιρεί. Η μέθοδος wdArray.exists() ελέγχει αν ένα αντικείμενο τύπου wdAsset υπάρχει στην wdArray. Η μέδοθος createSymbol δέχεται ένα αντικείμενο wdAsset και διαβάζοντας τα γραφικά του περιεχόμενα (χρησιμοποιώντας την property SVG), τα τοποθετεί σαν νέο στοιχείο στην περιοχή <def> του root SVGelement. Τα διαθέσιμα αντικείμενα πορτών ή παραθύρων στην βάση δεδομένων παρουσιάζονται σε μια λίστα από εικονίδια στην πλευρική μπάρα του editor σχεδίασης. Υπάρχει ένα στο html έγγραφο με id=”divWallProperties” του οποίου ο κώδικας σε γλώσσα html είναι ο εξής:

ΚΩΔΙΚΑΣ #21:

```
<div id="divWdList">
<span id="wdlTl">XXX</span> LIBRARY<br />
<div class="pallete" id="divWdlPallete">

</div>
<p> </p>
<table width="286" border="1">
  <tr>
    <td  width="92"  rowspan="5"  align="center"><img  id="wdlSample"
src="img/noTexture.jpg" width="40" height="40" /></td>
    <td width="178" colspan="3"><span id="wdlTitle">XXX</span></td>
  </tr>
  <tr>
    <td height="29" colspan="3"><span id="wdlDesc">XXX</span></td>
  </tr>
  <tr>
    <td height="14">WIDTH</td>
    <td>HEIGHT</td>
    <td>ELEV</td>
  </tr>
  <tr>
    <td              height="14"><span              id="wdlDesc"><span
id="wdlWidth">XXX</span></td>
    <td><span id="wdlDesc"><span id="wdlHeight">XXX</span></td>
    <td><span id="wdlDesc"><span id="wdlElev">XXX</span></td>
  </tr>
  <tr>
    <td          height="14"          colspan="3">Cost:          <span
id="wdlCost">XXX</span></td>
  </tr>
</table>
<p>
  <input     type="submit"     name="btnInsertWd"     id="btnInsertWd"
value="Insert" />
  <input     type="submit"     name="btnCancelWd"     id="btnCancelWd"
value="Cancel" />
```

```
</p>
</div>
```

Το παραπάνω είναι υπεύθυνο για την παρουσίαση όλων εκείνων των στοιχείων που υλοποιούν το τμήμα του user interface που αφορά την παρουσίαση της λίστας των διαθέσιμων παραθύρων ή πορτών. Παρατηρούμε ένα εμφωλευμένο div με id=”pallete” το οποίο θα παίξει τον ρόλο τις λίστας των εικονιδίων με τα αντικείμενα (πόρτες/παράθυρα). Επίσης παρατηρούμε ότι στο υπόλοιπο μέρος του div υπάρχει ένα html table element που περιλαμβάνει μαρκαρισμένα με id στοιχεία κειμένου , τα οποία διαδραστικά θα εμφανίζουν πληροφορίες για το αντικείμενο που επιλέγει ο χρήστης (π.χ. , κτλ). Ο κώδικας Javascript που υλοποιεί την διαδραστικότητα στο περιέχεται στο αρχείο windoorProperties.js και υλοποιείται συγκεκριμένα από την κλάση WdListCardClass. Η κλάση αυτή δημιουργεί τις απαραίτητες συνδέσεις (bindings) μεταξύ html στοιχείων και Javascript αντικειμένων. Ο κώδικας για την δήλωση της κλάσης είναι ο εξής:

ΚΩΔΙΚΑΣ #22:

```
1.  function WdListCardClass()
2.  {

3.  this.pallete = document.getElementById("divWdlPallete");

4.  this.wdlTl = document.getElementById("wdlTl");

5.  this.wdlSample = document.getElementById("wdlSample");
6.  this.wdlTitle = document.getElementById("wdlTitle");
7.  this.wdlDesc = document.getElementById("wdlDesc");
8.  this.wdlCost = document.getElementById("wdlCost");

9.  this.wdlWidth = document.getElementById("wdlWidth");
10. this.wdlHeight = document.getElementById("wdlHeight");
11. this.wdlElev = document.getElementById("wdlElev");

12. this.wdlSample = document.getElementById("wdlSample");

13. this.btnApply = document.getElementById("btnInsertWd");
14. this.btnCancel = document.getElementById("btnCancelWd");

15. this.btnCancel.addEventListener("click",clickWdlCancel,false);
16. this.btnApply.addEventListener("click",clickWdlInsert,false);


17. this.swatches = new Array();

18. this.selected = 1;
19. }
```

Παρατηρούμε ότι δημιουργούνται properties με αναφορές στα στοιχεία html του . Η συνάρτηση fillWdlSwatches() δέχεται μια μεταβλητή τύπου mode

η οποία καθορίζει αν θέλουμε να εμφανιστούνε τα παράθυρα ή οι πόρτες που είναι αποθηκευμένες στην βάση δεδομένων. Κατόπιν, χρησιμοποιώντας ασύγχρονη επικοινωνία με τον server, δέχεται με μορφή XML την λίστα με τα διαθέσιμα αντικείμενα και γεμίζει με αντιπροσωπευτικά εικονίδια την εμφωλευμένη .

Ο κώδικας της συνάρτησης είναι ο εξής:

ΚΩΔΙΚΑΣ #23:

```
1. function fillWdlSwatches(mode)
2. {
3. WdListCard.wdlTl.innerHTML = mode;

4. WdListCard.clearSwatches();

5. xmlhttp=new XMLHttpRequest();

6. xmlhttp.onreadystatechange=function()
7. {
8. if (xmlhttp.readyState==4 && xmlhttp.status==200)
9. {
10. Body.style.cursor = "default";
11. var res=xmlhttp.responseXML;

12. var tarray = res.getElementsByTagName("windoor");

13. var i ;

14. for (i=0;i<tarray.length;i++)
15. {
16. var swatch = document.createElement("img");

17. var el = tarray[i];
18. console.log(tarray[i]);
19. var A = {

20. "idd":el.getAttributeNS(null,"id"),
21. "src":wdPath + el.getAttributeNS(null,"img"),
22. "class":"swatch"
23. }

24. assign(swatch,A);

25. swatch.addEventListener("click",doUpdateWdlSwatch,false);

26. WdListCard.swatches.push(swatch);

27. WdListCard.pallete.appendChild(swatch);



28. }

29. udpateSelectedWdlSwatch(1);


30. }
31. }
32. if (mode=="window")
33. {
```

```
34. xmlhttp.open("GET","wdList.PHP?type='window'",true);
35. }
36. else
37. {
38. xmlhttp.open("GET","wdList.PHP?type='door'",true);
39. }
40. xmlhttp.send();
41. Body.style.cursor = "url(img/ajax.png) , auto";

42. }
```

Παρατηρούμε ότι αρχικά καλείται η μέθοδος `WdListCard.clearSwatches();` η οποία καθαρίζει το εσωτερικό του < div id= “pallete” > από οποιαδήποτε εναπομείναντα html elements. Κατόπιν δημιουργείται ένα νέο XMLHttpObject το οποίο αποτελεί τον πυρήνα της ασύγχρονης επικοινωνίας με τον server. Με την μέθοδο onreadyState του XMLHttpObject, καθορίζεται ο κώδικας που θα εκτελεστεί όταν το αντικείμενο XMLHttpRequest βρίσκεται σε κατάσταση ετοιμότητας (δεν βρίσκεται δηλ. «απασχολημένο» σε κατάσταση αποστολής ή λήψης δεδομένων). Η property onreadyState εξισώνεται με την δήλωση συνάρτησης επί τόπου (πράγμα που αποτελεί χαρακτηριστικό της Javascript), στην οποία περιλαμβάνεται ο κώδικας που θέλουμε να εκτελεστεί. Ο κώδικας αυτός ελέγχει κατά την κατάσταση ετοιμότητας την property readyState η οποία θα πρέπει να είναι ίση με 4, δηλαδή ότι η σύνδεση με τον διακομιστή βρίσκεται σε κατάσταση, στην οποία έχει ολοκληρωθεί η αποστολή δεδομένων. Επιπλέον ελέγχει την property status, η οποία θα πρέπει να έχει την τιμή 200, που σημαίνει ότι η αποστολή δεδομένων από τον διακομιστή πραγματοποιήθηκε με επιτυχία. Εφόσον η επικοινωνία έχει ολοκληρωθεί, τοποθετείται η απάντηση του διακομιστή, υπό μορφή xml, στην μεταβλητή res. Να σημειωθεί ότι η μεταβλητή res δεν περιλαμβάνει απλώς ένα αλφαριθμητικό σε γλώσσα XML αλλά ένα ολόκληρο έγγραφο XML DOM, με root element και εμφωλευμένα nodes, ανάλογα με την μορφή της απάντησης.

Τα nodes που χρειάζονται είναι τα <windoor>, τα οποία επιστρέφονται σε μορφή λίστας με την κλήση της συνάρτησης res.getElementsByTagName(“windoor”). Με βρόγχο εξετάζεται κάθε ένα xml element τύπου windoor και δημιουργούνται τα κατάλληλα html elements (<img>) για την αναπαράσταση του στην ιστοσελίδα. Το υπόλοιπο σώμα της συνάρτησης fillWdlSwatches() περιλαμβάνει την έναρξη της επικοινωνίας με τον server. Αυτό επιτυγχάνεται καλώντας την μέθοδο .open(). Το script της PHP το οποίο αναλαμβάνει την επικοινωνία με την βάση δεδομένων είναι το wdList.PHP:

ΚΩΔΙΚΑΣ #24:

```
1. <?PHP
2. header('Content-Type: text/xml');

3. $tid = $_GET["type"];
```

```
4. if(!$dbconnect = mysql_connect('localhost', 'root', '')) {
5. echo "Connection failed to the host 'localhost'.";
6. exit;
7. } // if
8. if (!mysql_select_db('homedraw')) {
9. echo "Cannot connect to database 'test'";
10. exit;
11. } // if

12. $table_id = 'windoors';
13. $query = "SELECT id,title,img FROM $table_id WHERE type = $tid";
14. $dbresult = mysql_query($query, $dbconnect);


15. // create a new XML document
16. $doc = new DomDocument('1.0');




17. // add root node
18. $root = $doc->createElement('root');
19. $root = $doc->appendChild($root);

20. // process one row at a time
21. while($row = mysql_fetch_assoc($dbresult)) {

22. // add node for each row
23. $child = $doc->createElement("windoor");
24. $newnode = $root->appendChild($child);

25. // add a child node for each field
26. foreach ($row as $fieldname => $fieldvalue) {


27. $newnode->setAttribute($fieldname,$fieldvalue);



28. } // foreach
29. } // while
30. // get completed xml document
31. $xml_string = $doc->saveXML();
32. echo $xml_string;
33. ?>
```

Καταρχήν με την εκτέλεση της εντολής header ρυθμίζεται ότι ο τύπος του παραγόμενου εγγράφου (αφού εκτελεστεί η PHP) θα είναι μορφής xml. Κατόπιν, ελέγχεται αν ήρθε μεταβλητή με την ονομασία type από τον φυλλομετρητή και τι τιμή περιλαμβάνει αυτή. Καλείται η mysql_connect για να γίνει σύνδεση με τον mysql server. Εκτελείται η mysql_select_db με όρισμα το όνομα της βάσης δεδομένων της εφαρμογής που είναι το “homedraw”.Με βάση το type, εκτελείται το query που διαλέγει της εγγραφές εκείνες του πίνακα windoors που το type τους ισούται με την τιμή της μεταβλητής type που έστειλε ο φυλλομετρητής.

Χρησιμοποιείται η εντολή `$doc = new DomDocument('1.0');` για να δημιουργηθεί ένα νέο έγγραφο XML στο οποίο, με τις συναρτήσεις createElement() και appendChild(), δημιουργούνται nodes για κάθε εγγραφή που ανακτάται από την βάση δεδομένων. Για κάθε στήλη της εγγραφής τοποθετείται, με την setProperty(), μια

αντίστοιχη ιδιότητα στο συγκεκριμένο node. Έτσι σιγά-σιγά χτίζεται η xml δομή που μεταφέρει τα δεδομένα που διαβάστηκαν από την βάση.

Η εντολή $xml_string = $doc->saveXML(); μετατρέπει την δενδρική DOM δομή που κατασκευάστηκε, σε ένα απλό αλφαριθμητικό το οποίο αποθηκεύεται στην μεταβλητή xml_string. Το μόνο που μένει είναι η PHP να τα κάνει echo τα περιεχόμενα της μεταβλητής xml_string ώστε να τυπωθούν στο τρέχων έγγραφο το οποίο θα επιστραφεί στον φυλλομετρητή ως XML document. Έτσι γίνεται η επικοινωνία μεταξύ φυλλομετρητή και διακομιστή και βάσης δεδομένων ή συγκεκριμένα μεταξύ Javascript και PHP με mysql.

Επιστρέφοντας στην κλάση αντικειμένων wdAsset, παρατηρούμε ότι η μέθοδος wdAsset.load() χρησιμοποιεί το XMLHttpObject για να επικοινωνήσει με τον server και να φορτώσει τα κατάλληλα δεδομένα στα properties του νέου αντικειμένου, τύπου wdAsset, που δημιουργείται:

ΚΩΔΙΚΑΣ 25:

```
1.  WdAsset.prototype.load = function(tid)
2.  {

3.  xmlhttp=new XMLHttpRequest();
4.  xmlhttp.open("GET","wdlDet.PHP?wdlId=" + tid,false);
5.  xmlhttp.send("");

6.  var xmlRes=xmlhttp.responseXML;



7.  var el = xmlRes.getElementsByTagName("windoor")[0];

8.  this.id = el.getAttributeNS(null,"id");
9.  this.img = el.getAttributeNS(null,"img");

10. this.w = el.getAttributeNS(null,"width");
11. this.h = el.getAttributeNS(null,"height");
12. this.e = el.getAttributeNS(null,"elevate");
13. this.type = el.getAttributeNS(null,"type");
14. this.title = el.getAttributeNS(null,"title");
15. this.desc = el.getAttributeNS(null,"description");
16. this.cost = el.getAttributeNS(null,"cost");
17. this.X3D = el.getAttributeNS(null,"X3D");
18. this.SVG = el.getAttributeNS(null,"SVG");
19. //load the SVG

20. xmlhttp.open("GET",wdPath + this.SVG ,false);
21. xmlhttp.send("");

22. var xmld = xmlhttp.responseXML;

23. var el = xmld.getElementById("content");
24. console.log(xmld);
25. this.def = el;
26. }
```

Παρατηρούμε ότι στην μέθοδο αυτή δημιουργείται ένα XMLHttpObject το οποίο, χρησιμοποιώντας την μέθοδο .open(), με παραμέτρους “GET” και καλώντας το script wdlDet.PHP, λαμβάνει σε μορφή XML απάντησης όλα τα πεδία των δεδομένων για ένα αντικείμενο windoor που είναι αποθηκευμένο στην βάση. Κατόπιν τα καταχωρεί αντίστοιχα σαν properties στο νέο wdAsset object που έχει δημιουργηθεί. Επίσης χρησιμοποιεί την μέθοδο .open() για να διαβάσει ένα αρχείο SVG (το οποίο στην ουσία είναι XML τύπου αρχείο) από το filesystem του διακομιστή ώστε να φορτώσει τα γραφικά του wdAsset σαν στοιχείο στο <def> του root SVGelement.

Ο κώδικας του αρχείου wdlDet.PHP είναι παρόμοιος με εκείνον του αρχείου wdList.PHP που παρουσιάσαμε παραπάνω:

ΚΩΔΙΚΑΣ #26:

```
<?PHP
header('Content-Type: text/xml');

$tid = $_GET["wdlId"];


if(!$dbconnect = mysql_connect('localhost', 'root', '')) {
echo "Connection failed to the host 'localhost'.";
exit;
} // if
if (!mysql_select_db('homedraw')) {
echo "Cannot connect to database 'test'";
exit;
} // if
$table_id = 'windoors';
$query = "SELECT * FROM $table_id WHERE id = $tid" ;
$dbresult = mysql_query($query, $dbconnect);
// create a new XML document
$doc = new DomDocument('1.0');
// add root node
$root = $doc->createElement('root');
$root = $doc->appendChild($root);

// process one row at a time
while($row = mysql_fetch_assoc($dbresult)) {

// add node for each row
$child = $doc->createElement("windoor");
$newnode = $root->appendChild($child);

// add a child node for each field
foreach ($row as $fieldname => $fieldvalue) {
$newnode->setAttribute($fieldname,$fieldvalue);
} // foreach
} // while
// get completed xml document
$xml_string = $doc->saveXML();
echo $xml_string;
?>
```

Με παρόμοιο τρόπο λειτουργούν όλα τα αντικείμενα που απαιτούν assets, όπως π.χ. οι υφές για τις οποίες χρησιμοποιούνται οι κλάσεις textureCardClass() ή τα

αντικείμενα διακόσμησης για τα οποία χρησιμοποιούνται οι κλάσεις decoAsset και decoProperties που περιγράφονται στα αρχεία Javascript textureProperties.js και decoAsset.js και decoProperties.js αντίστοιχα. Επίσης τα αντίστοιχα PHP scripts που επικοινωνούν με την βάση δεδομένων είναι τα textureList.PHP (το οποίο επιστρέφει μια λίστα με όλες τις υφές) , textureDet.PHP (το οποίο επιστρέφει πληροφορίες για μια συγκεκριμένη υφή), decoList.PHP (το οποίο επιστρέφει λίστα με όλα τα αντικείμενα διακόσμησης) και decoDet.PHP (το οποίο επιστρέφει πληροφορίες σχετικά με ένα συγκεκριμένο αντικείμενο διακόσμησης).

5.5 Υλοποίηση δομών διαχείρισης χρωμάτων

Σε διάφορα τμήματα της εφαρμογής απαιτείται διαχείριση χρωμάτων. Για τον σκοπό αυτό και για να αποφευχθεί η επαναλαμβανόμενη συγγραφή παρόμοιου κώδικα, δημιουργήθηκαν δομές και συναρτήσεις οι οποίες βοηθούν στην διαχείριση αυτή. Η πιο βασική από όλες είναι η κλάση Color η οποία δημιουργεί αντικείμενα τα οποία αποθηκεύουν πληροφορία για ένα συγκεκριμένο χρώμα:

ΚΩΔΙΚΑΣ #27:

```
1.  function Color(r,g,b)
2.  {
3.  this.r=r;
4.  this.g=g;
5.  this.b=b;
6.  this.h = 0;
7.  this.s = 0;
8.  this.v = 0;
9.  this.validateRGB();
10. this.toHSV();
11. }
```

Παρατηρούμε, ότι η κλάση περιλαμβάνει 3 properties r,g,b τα οποία περιέχουν τιμές για την περιγραφή του χρώματος με το χρωματικό μοντέλο RGB (Red Green Blue). Το χρωματικό μοντέλο RGB καθορίζει ένα χρώμα με βάση τρεις μεταβλητές οι οποίες περιγράφουν το ποσοστό των τριών βασικών χρωμάτων που αναμιγνύονται: Κόκκινο, Πράσινο και Μπλε. Οι τιμές που παίρνουν οι τρείς αυτές μεταβλητές κυμαίνονται από 0 έως 250. Η τριάδα (r,g,b) = (0,0,0) περιγράφει το απόλυτο μαύρο και η τριάδα (r,g,b) = (250,250,250) περιγράφει το απόλυτο λευκό. Τα άλλα τρία properties h,s,v περιέχουν τιμές για την περιγραφή του χρώματος με βάση το χρωματικό μοντέλο HSV (Hue Saturation Volume). Το hue περιλαμβάνει αριθμητική τιμή που αντιστοιχεί στην χρωματική χροιά, το saturation στην ένταση και το volume στην φωτεινότητα του χρώματος.

Παρακάτω θα αναφερθούν μερικές από τις πιο σημαντικές μεθόδους που υλοποιήθηκαν για την κλάσης color.

i) Color.setRGB(r,g,b)

Διοχετεύει μια νέα χρωματική τιμή στο αντικείμενο υπό μορφή rgb.

ii) Color.setHSV(h,s,v)

Διοχετεύε μια νέα χρωματική τιμή στο αντικείμενο υπό μορφή hsv.

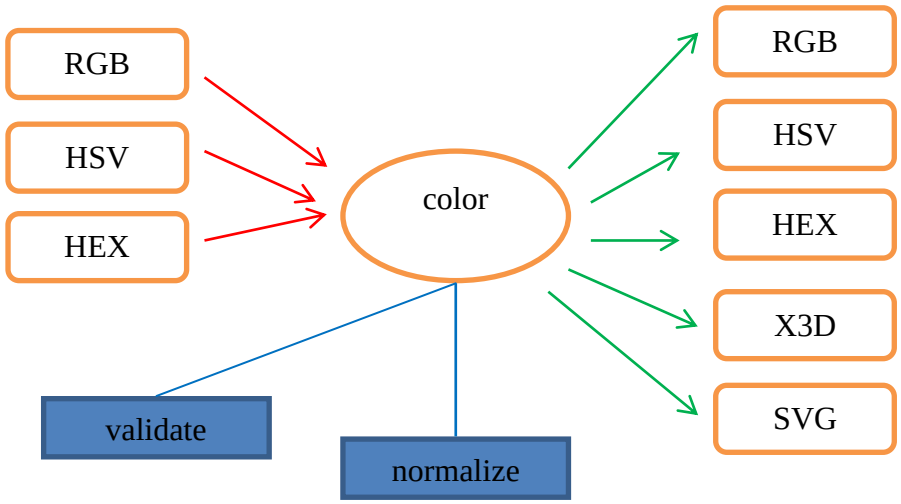
iii) Color.setHex(hex)

Διοχετεύε μια νέα χρωματική τιμή στο αντικείμενο υπό δεκαεξαδική μορφή.

iv) Color.getColor(color)

Η συγκεκριμένη μέθοδος δέχεται ένα άλλο αντικείμενο τύπου color και αντιγράφει στο παρόν τις τιμές χρώματος.

Επίσης υλοποιήθηκαν και μέθοδοι μετατροπής χρωμάτων από την μια μορφή στην άλλη, όπως από HSV σε RGB, με την μέθοδο Color.toRGB(). Παρακάτω συνοψίζονται γραφικά όλες οι δυνατότητες της κλάσης Color:

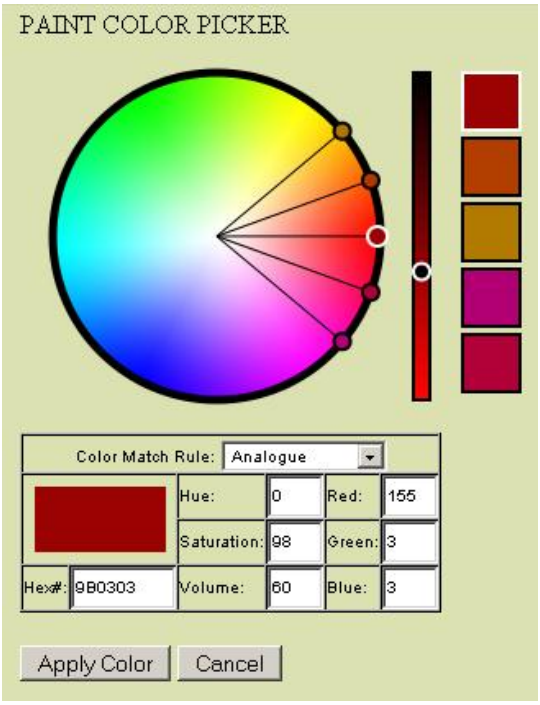


Εικόνα 96 – Γραφική αναπαράσταση των δυνατοτήτων της κλάσης color

5.6 Υλοποίηση του επιλογέα χρωμάτων (color picker)

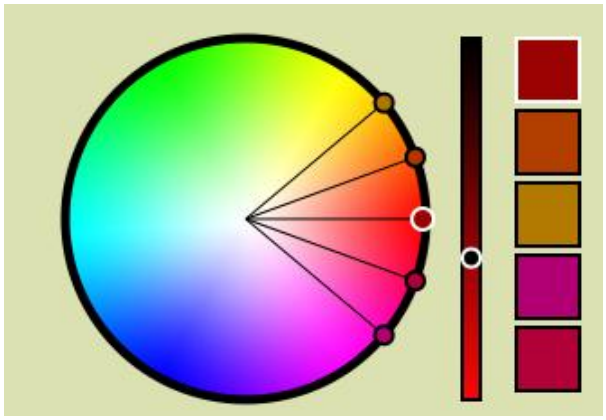
Όπως αναφέρθηκε και στο προηγούμενο κεφάλαιο, η εφαρμογή περιλαμβάνει ένα ισχυρό σύστημα επιλογής χρωμάτων αλλά και δημιουργίας χρωματικών ομάδων που μοιράζονται μια αισθητική αρμονία. Η υλοποίηση του επιλογέα χρωμάτων χρησιμοποιώντας στοιχεία SVG και html όσον αφορά την διεπαφή του, και κώδικα

Javascript όσον αφορά την λογική του. Το interface του color picker παρουσιάζεται στην παρακάτω εικόνα:



Εικόνα 97 – Το interface του επιλογέα χρωμάτων

Το κεντρικό αρχείο Javascript που περιέχει το μεγαλύτερο μέρος του κώδικα που υλοποιεί τον επιλογέα χρωμάτων είναι το colorpick.js Ένα κομμάτι του interface υλοποιείται αποκλειστικά με SVG γραφικά και είναι το εξής:



Εικόνα 98 – Τα SVG στοιχεία του interface του επιλογέα χρωμάτων

Η συνάρτηση initColorPicker αρχικοποιεί το περιβάλλον του επιλογέα χρωμάτων κατασκευάζοντας πολλά από τα στοιχεία του gui του.

Η κλάση που υλοποιεί τον επιλογέα χρωμάτων σαν αντικείμενο είναι η Cpicker:

ΚΩΔΙΚΑΣ #28:

```
1.  function CPicker()
2.  {
3.  this.list = null;
4.  this.state = "analogue";
5.  this.delta = 20;
6.  this.base = 0;
7.  this.selected = null;
8.  this.selSwatch = null;
9.  this.swatches = null;
10. }
```

Η κλάση περιέχει της εξής ιδιότητες:

- **List:** περιλαμβάνει array με όλα τα handles επιλογής χρωμάτων (5 στον αριθμό).
- **State:** δηλώνει σε πια κατάσταση επιλογής αρμονικών χρωμάτων βρισκόμαστε, συγκεκριμένα στην τρέχουσα φόρμουλα επιλογής χρωμάτων.
- **Delta:** η γωνιακή απόσταση μεταξύ 2 συνεχόμενων handles επιλογής.
- **Base:** η γωνία του βασικού handle επιλογής χρωμάτων.
- **Selected:** Ποιο είναι το τρέχον επιλεγμένο handle.
- **selSwatch:** Ποιο είναι το τρέχον επιλεγμένο swatch χρώματος.
- **swatches:** Λίστα με τα 5 swatches χρωμάτων.

Το κάθε handle επιλογής χρώματος (η μορφή του οποίου εμφανίζεται στην παρακάτω εικόνα) υλοποιείται ως αντικείμενο στην Javascript χρησιμοποιώντας την κλάση colHandle.

Εικόνα 99 – Αναπαράσταση ενός color handle

Ο κώδικας της colHandle παρουσιάζεται παρακάτω:

ΚΩΔΙΚΑΣ #29:

```
1.  function ColHandle(i,x,y)
2.  {
3.  this.index = i;
4.  this.x=x+cOrigin.x;
5.  this.y= -y + cOrigin.y;
6.  this.SVG;
7.  this.lSVG;
8.  this.volume = 70;
9.  this.draw();
10. }
```

Η συνάρτηση κατασκευής της δέχεται 3 ορίσματα:

- **i:** για να ορίσει το index του συγκεκριμένου handle.
- **x,y** για να ορίσει τις συντεταγμένες του συγκεκριμένου handle στο επίπεδο του χρωματικού κύκλου.

Επίσης περιλαμβάνονται τα εξής properties:

- **index:** η αρίθμηση του handle (για να το ξεχωρίζουμε ανάμεσα στα 5).
- **x:** συντεταγμένη x του Handle.
- **y:** συντεταγμένη y του Handle.
- **SVG:** αναφορά στο γραφικό στοιχείο <circle> που αναπαριστά το handle.
- **lSVG:** αναφορά στο γραφικό στοιχείο <line> που αποτελεί τμήμα της γραφικής αναπαράστασης του handle και είναι η γραμμή που ενώνει τον κύκλο του εκάστοτε handle με το κέντρο του χρωματικού κύκλου.
- **Volume:** αναπαριστά το ποσοστό της φωτεινότητας του χρώματος που έχει επιλεγεί από το συγκεκριμένο handle.

Ακολουθεί η περιγραφή μερικών από τις μεθόδους της κλάσης colHandle.

i) colHandle.draw()

Αποτελεί την μέθοδο η οποία δημιουργεί τα απαραίτητα SVG στοιχεία για την γραφική αναπαράσταση του color handle. Τα βασικά στοιχεία που δημιουργούνται είναι ένας κύκλος (<circle> element) και μια γραμμή (<line> element).

ii) colHandle.getR()

Η μέθοδος επιστρέφει την ακτίνα του συγκεκριμένου color Handle:

ΚΩΔΙΚΑΣ #30:

```
1. ColHandle.prototype.getR = function()
2. {
      a. return calcDistance(this.x,this.y,cOrigin.x,cOrigin.y);
3. }
```

iii) colHandle.getAngle()

Η μέθοδος επιστρέφει την γωνία του συγκεκριμένου color Handle χρησιμοποιώντας την βοηθητική μέθοδο getAngleFromOrigin();

ΚΩΔΙΚΑΣ #31:

```
1. ColHandle.prototype.getAngle = function()
2. {
3. var tx = this.getPx();
4. var ty = this.getPy();

5. var a = getAngleFromOrigin(tx,ty);

6. return a;

7. }


8. function getAngleFromOrigin(x,y)
9. {
10.var tx = x;
11.var ty = y;

12.var a;

13.// Find the Polar Coordinate Angle;

14.if (tx>0 && ty>=0)
15.{
16.a = Math.atan(ty/tx);
17.}
18.else if (tx > 0 && ty < 0)
19.{
20.a = Math.atan(ty/tx) + (2 * Math.PI);
21.}
22.else if (tx < 0)
23.{
24.a = Math.atan(ty/tx) + Math.PI;
25.}
26.else if (tx == 0 && ty > 0)
27.{
28.a = Math.PI / 2 ;
29.}
30.else if (tx == 0 && ty < 0)
31.{
32.a = 3 * (Math.PI / 2) ;
33.}
34.else if (tx == 0 && ty == 0)
35.{
36.a = 0 ;
37.}

38.// Now convert the a to degrees;

39.a = radToDeg(a);

40.return a;
41.}
```

---

iv) colHandle.move(x,y)

Η συγκεκριμένη μέθοδος πραγματοποιεί μετακίνηση του color Handle μέσα σε ένα σημείο του χρωματικού κύκλου. Χρησιμοποιεί την βοηθητική μέθοδο moveTrue(x,y)

ΚΩΔΙΚΑΣ #32:

```
1. ColHandle.prototype.moveTrue = function(x,y)
2. {
3. this.x = x;
4. this.y = y;
5. this.SVG.setAttributeNS(null,"transform","translate("+ x + "," + y +
   ")");

6. var A = {
7. "x1": this.x,
8. "y1": this.y,
9. "x2": cOrigin.x,
10. "y2": cOrigin.y
11. }

12. assign(this.lSVG,A);

13. this.fill();


14. }

15. ColHandle.prototype.move = function(x,y)
16. {
17. var nx = x + cOrigin.x;
18. var ny = -y + cOrigin.y;

19. this.moveTrue(nx,ny);

20. }
```

---

vi) colHandle.movePolar(r,a)

Η συγκεκριμένη μέθοδος πραγματοποιεί μετακίνηση του color Handle σε μια συγκεκριμένη γωνία και απόσταση από το κέντρο του χρωματικού κύκλου.

ΚΩΔΙΚΑΣ #33:

```
1. ColHandle.prototype.movePolar = function(r,a)
2. {
3. var angle = degToRad(a);
4. var nx = r * Math.cos(angle);
5. var ny = r * Math.sin(angle);
6. this.move(nx,ny);
7. }
```

Παρατηρούμε, ότι η συνάρτηση χρησιμοποιώντας τριγωνομετρία, κάνει μετατροπή των πολικών συντεταγμένων (γωνία a και ακτίνα r) σε καρτεσιανές (x,y) και κατόπιν καλεί την move(x,y) για να μετακινηθεί το color Handle.

Επιπλέον υλοποιήθηκε και η μέθοδος colHandle.moveByRgb(r,g,b) η οποία δέχεται μια χρωματική τιμή rgb και μετακινεί το color Handle σε εκείνο το σημείο του χρωματικού κύκλου που αναπαριστάται το συγκεκριμένο χρώμα.

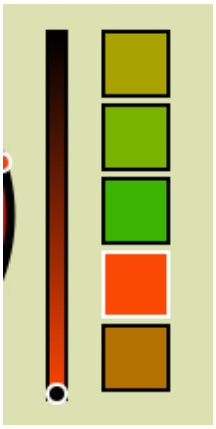
Ο κώδικας παρουσιάζεται παρακάτω:

ΚΩΔΙΚΑΣ #34:

```
1. ColHandle.prototype.moveByRGB = function(r,g,b)
2. {
3. var col = new Color(r,g,b);
4. this.setVolume(col.v);
5. var r = col.s;
6. var a = col.h;
7. this.movePolar(r,a);
8. }
```

Στην ουσία χρησιμοποιεί τις γεωμετρικές ιδιότητες που έχει η αναπαράσταση του μοντέλου HSV πάνω στον χρωματικό κύκλο: το hue αποτελεί την γωνία και το saturation την απόσταση από το κέντρο του κύκλου.

Τέλος στο αρχείο colorPick.js περιλαμβάνεται και η υλοποίηση της κλάσης cSwatch η οποία ελέγχει τα 5 swatches χρωμάτων δίπλα από τον χρωματικό κύκλο (εικόνα 100):

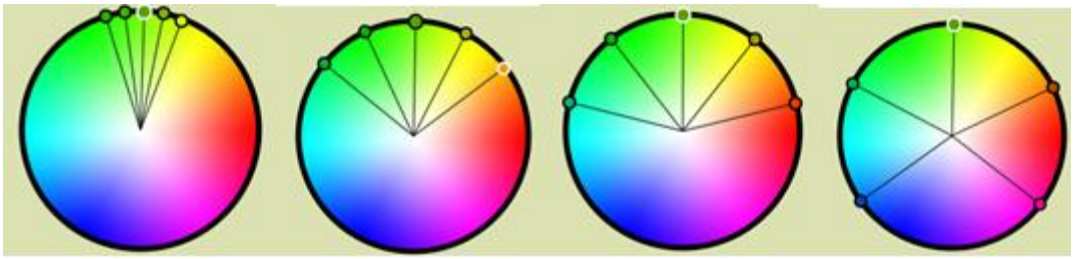


Εικόνα 100 – Η μπάρα φωτεινότητας και τα 5 color swatches

Επιστρέφοντας τώρα στο αντικείμενο του τύπου colPick, που περιγράφει ολόκληρο τον επιλογέα χρωμάτων, περιλαμβάνονται κάποιες πολύ βασικές μέθοδοι οι οποίες υλοποιούν τις φόρμουλες επιλογής αρμονικών χρωμάτων. Συγκεκριμένα υπάρχει η μέθοδος colPick.changeHandle(index) η οποία καλείται, όταν παρατηρείται αλλαγή στην θέση οποιουδήποτε color Handle, ούτως ώστε να μετακινηθούν ανάλογα και τα άλλα σύμφωνα με την φόρμουλα που βρίσκεται σε ισχύ. Όπως αναφέρθηκε και στο προηγούμενο κεφάλαιο υλοποιήθηκαν 4 φόρμουλες επιλογής αρμονικών χρωμάτων.

Αναλογική

Η πρώτη από αυτές είναι η αναλογική η οποία βασίζεται στο βασικό color Handle και απαιτεί από τα υπόλοιπα να διαιρούν τον χώρο δεξιά και αριστερά του σε ίσες γωνίες όπως δείχνουν οι εικόνες:

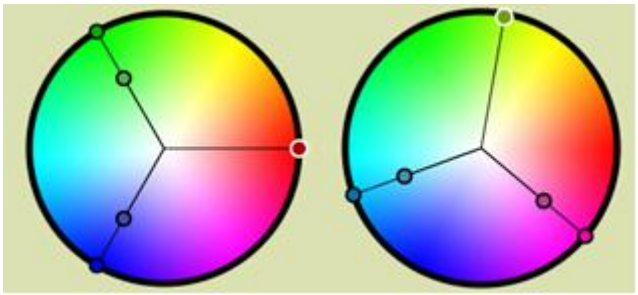


Εικόνα 101 – Παραδείγματα από την χρήση του color picker με αναλογική φόρμουλα επιλογής

Για να υλοποιηθεί αυτή η συμπεριφορά, η changeHandle καλεί την changeAnalogue και τις διοχετεύει το index του handle που μετακινήθηκε. Η changeAnalogue στην ουσία καλεί την μέθοδο subAngles() (η οποία βρίσκει την διαφορά δύο γωνιών σε μοίρες) για να υπολογίσει την property delta του color picker, η οποία στην ουσία είναι η απόσταση σε μοίρες δύο συνεχόμενων color Handles. Γνωρίζοντας και την γωνία της βάσης, αρχίζει σιγά σιγά και με πολικές συντεταγμένες να επανατοποθετεί τα υπόλοιπα color Handles σε αντίστοιχες γωνίες από το βασικό, διατηρώντας μεταξύ τους τον κανόνα αρμονίας.

Τριαδική

Η δεύτερη φόρμουλα επιλογής χρωμάτων είναι η τριαδική η οποία επιτρέπει τα colorHandles να έχουν σταθερή απόσταση μεταξύ τους 120 μοιρών. Δηλαδή τα color Handles θα βρίσκονται πάντα σε τρεις ευθείες που χωρίζουν τον κύκλο σε 3 ίσα τμήματα όπως δείχνουν οι παρακάτω εικόνες:

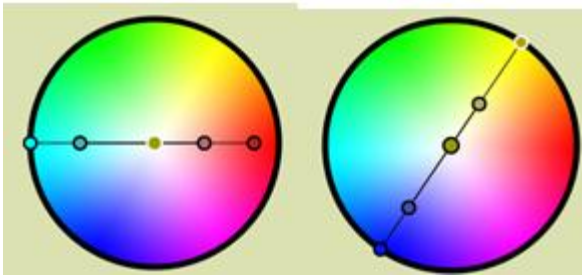


Εικόνα 102 - Παραδείγματα από την χρήση του color picker με τριαδική φόρμουλα επιλογής

Για να υλοποιηθεί η συγκεκριμένη φόρμουλα, η changeHandle() καλεί την changeTriad() διοχετεύοντάς της το index του Handle που κινήθηκε. Το delta τώρα είναι σταθερό σε 120 και η μέθοδος προσπαθεί να καθορίσει την γωνία του βασικού color Handle. Από εκεί και πέρα τοποθετεί τα υπόλοιπα σε αποστάσεις 120 μοιρών, δεξιά και αριστερά, χρησιμοποιώντας πολικές συντεταγμένες και καλώντας τις αντίστοιχες μεθόδους colHandle.movePolar(r,a) κτλ.

Συμπληρωματική Φόρμουλα

Στην συμπληρωματική φόρμουλα τα color Handles σχηματίζουν μεταξύ τους γωνία 180 μοιρών το οποίο ουσιαστικά σημαίνει, ότι όλα τοποθετούνται πάνω σε μια διάμετρο του χρωματικού κύκλου όπως δείχνουν και οι παρακάτω εικόνες:



Εικόνα 103 - Παραδείγματα από την χρήση του color picker με συμπληρωματική φόρμουλα επιλογής

Για να υλοποιηθεί αυτή η φόρμουλα επιλογής, η μέθοδος changeHandle() καλεί την changeComplementary() στην οποία διοχετεύει το index του color Handle που κινήθηκε. Ο κώδικας της changeComplementary() είναι παρόμοιος με εκείνον της changeTriad(), με την μόνη διαφορά ότι τώρα το property delta που περιγράφει την γωνία μεταξύ δύο συνεχόμενων color Handles ισούται με 180 μοίρες. Όπως και στην προηγούμενη φόρμουλα έτσι και εδώ καθορίζεται η γωνία του βασικού color Handle και κατόπιν τοποθετούνται τα άλλα σε γωνία ταυτισμένη με αυτό ή 180 μοίρες αντιδιαμετρικά του.

Μονοχρωματική Φόρμουλα

Η τελική φόρμουλα είναι και η απλούστερη καθώς παράγει αρμονικά χρώματα της ίδιας χροιάς αλλά διαφορετικής έντασης και φωτεινότητας. Τα color Handles βρίσκονται στην ίδια ακτίνα έχοντας όλα την ίδια γωνία. Για να επιτευχθεί αυτό, η changeHandle() καλεί την changeMono() διοχετεύοντας το index του color Handle που μετακινήθηκε. Το delta τώρα έχει την τιμή 0. Παρόμοια με τις άλλες μεθόδους, πρώτα καθορίζεται η γωνία του βασικού color Handle και ύστερα, όλα τα υπόλοιπα (χρησιμοποιώντας βρόγχο) ταυτίζονται στον ίδιο προσανατολισμό με αυτό. Επίσης, συνοψίζοντας πρέπει να σημειωθεί ότι η changeHandle() εξετάζει την property colHandle.state για να δει πια από τις 4 μεθόδους θα καλέσει.

5.7 Υλοποίηση διαδικασιών μετάφρασης δισδιάστατης γεωμετρίας σε τρισδιάστατη

Το δεύτερο, πιο σημαντικό τμήμα της εφαρμογής, μετά την σχεδίαση της δισδιάστατης κάτοψης, είναι η μετατροπή της, σε τρισδιάστατη αναπαράσταση. Στην ουσία αναλύεται διεξοδικά το δισδιάστατο σχέδιο και μεταφράζεται σε τρισδιάστατη γεωμετρία. Σε επίπεδο Javascript γίνεται ανάλυση όλων των δομών που περιγράφουν τα αντικείμενα του δισδιάστατου σχεδίου και χρησιμοποιούνται κατάλληλες συναρτήσεις και κλάσεις αντικειμένων, ώστε να παραχθεί η τρισδιάστατη γεωμετρία. Όλη αυτή η μετατροπή υλοποιείται στο αρχείο export.js. Όταν ολοκληρωθεί η διαδικασία, στην ουσία θα έχει παραχθεί ένα αρχείο X3D, το οποίο περιγράφει όλη την τρισδιάστατη γεωμετρία της σκηνής και το μόνο που μένει είναι να φορτωθεί με το κατάλληλο plugin στον φυλλομετρητή ώστε ο χρήστης να ξεκινήσει να περιηγείται στον τρισδιάστατο χώρο που σχεδίασε. Το αρχείο export.js περιλαμβάνει

συναρτήσεις οι οποίες αναλαμβάνουν να δημιουργήσουν τις κατάλληλες δομές για την συγκρότηση ενός αρχείου τύπου X3D (το οποίο είναι αρχείο τύπου XML). Κατόπιν υπάρχουν συναρτήσεις που υλοποιούν δημιουργία X3D nodes όπως box, extrusion κτλ διοχετεύοντας τους βασικές παραμέτρους. Υπάρχουν και κλάσεις που υλοποιούν δομές low level τρισδιάστατης γεωμετρίας όπως vertices faces και meshes τα οποία στην γλώσσα X3D δηλώνονται ως παράμετροι των στοιχείων <IndexedFaceset>. Επίσης, υπάρχουν διαδικασίες που αποδίδουν υφές στα μοντέλα τρισδιάστατης γεωμετρίας ή και ακόμα επιβάλλουν κατάλληλους γεωμετρικούς μετασχηματισμούς. Επιπλέον, τοποθετούν αναφορές προς γεωμετρίες που περιγράφονται σε εξωτερικά αρχεία. Η βασική συνάρτηση που υλοποιεί όλη την μετατροπή καλώντας επιμέρους συναρτήσεις και μεθόδους είναι η testXML() η οποία περιγράφεται παρακάτω:

ΚΩΔΙΚΑΣ #35:

```
1. function testXML()
2. {
3. XMLdoc = createXMLdoc(document);
4. var i;
5. for (i in Walls.list) {
6. exportWall(XMLdoc, Walls.list[i]);
7. exportTapestry(Walls.list[i],XMLdoc);
8. }
9. for (i in Handles.list)
10. {
11. exportFills(XMLdoc,Handles.list[i]);
12. }
13. for (i in Floors.list)
14. {
15. exportFloors(XMLdoc,Floors.list[i]);
16. exportFloorTexture(XMLdoc,Floors.list[i]);
17. exportCeilingTexture(XMLdoc,Floors.list[i]);
18. }
19. exportViewPoint(XMLdoc);
20. var xmlString = (new XMLSerializer()).serializeToString(XMLdoc);
21. return xmlString;
22. }
```

Δηλώνεται μια μεταβλητή που αντιπροσωπεύει το xml έγγραφο, ονόματι XMLdoc, η οποία χρησιμοποιείται ως εξής:

```
XMLdoc = createXMLdoc(document);
```

Η συνάρτηση createXMLDoc() δημιουργεί ένα νέο XML DOM για το αρχείο X3D που θα δημιουργηθεί.

ΚΩΔΙΚΑΣ #36:

```
1. function createXMLdoc()
2. {
```

```
3.  var xml = document.implementation.createDocument("","X3D", null);

4.  var rt = xml.getElementsByTagName("X3D")[0];

5.  var sc = xml.createElement("Scene");

6.  rt.appendChild(sc);

7.  rt.setAttribute("profile","immersive");
8.  rt.setAttribute("version","3.2");


9.  return xml;
10. }
```

Για την δημιουργία του DOM το οποίο θα αποθηκευτεί στην μεταβλητή xml, η οποία επιστρέφεται, χρησιμοποιείται η εντολή:

```
var xml = document.implementation.createDocument("","X3D", null);
```

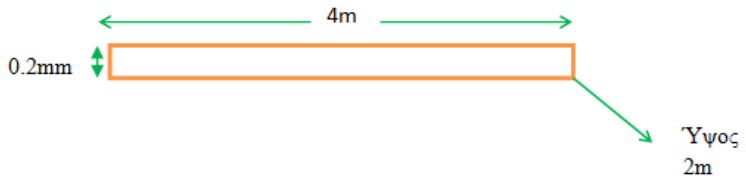
Σαν δεύτερο όρισμα ορίζεται το όνομα του root element του αρχείου XML, το οποίο στην περίπτωση αυτή είναι X3D.Κατόπιν δημιουργείται αναφορά στο στοιχείο X3D και δημιουργείται το αμέσως επόμενο βασικό στοιχείο ενός αρχείου X3D το οποίο είναι το <scene> Ρυθμίζονται οι παράμετροι profile:immersive και version: 3.2 για το X3D element.

5.7.1 Εξαγωγή τοίχων σε X3D

Παρατηρούμε στην textXML() πώς ακολουθεί βρόγχος για κάθε Wall object στο Walls.list:

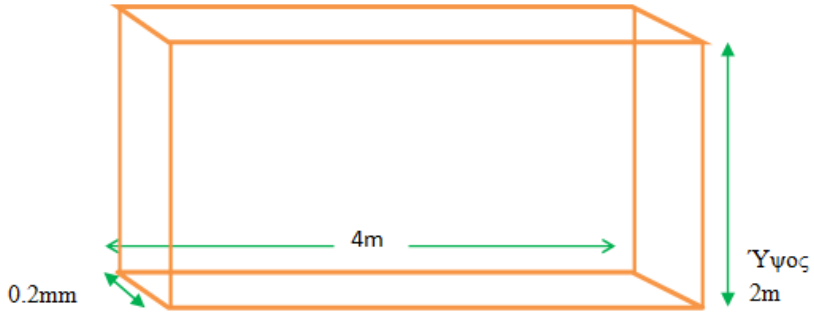
```
for (i in Walls.list) {
            exportWall(XMLdoc, Walls.list[i]);
            exportTapestry(Walls.list[i],XMLdoc);
      }
```

Στον οποίο καλούνται, για κάθε αντικείμενο τύπου wall, οι συναρτήσεις exportWall() και exportTapestry(). Ένας τοίχος, στο δισδιάστατο σχέδιο, πιθανόν να περιέχει οπές στις οποίες τοποθετούνται παράθυρα και πόρτες. Για να δημιουργηθεί η τρισδιάστατη γεωμετρία του τοίχου χρησιμοποιήθηκε το στοιχείο <box> σε X3D το οποίο δημιουργεί ένα τρισδιάστατο παραλληλόγραμμο καθορίζοντας ύψος, μήκος, πλάτος. Έστω ότι έχουμε σε δισδιάστατη μορφή τον παρακάτω τοίχο:



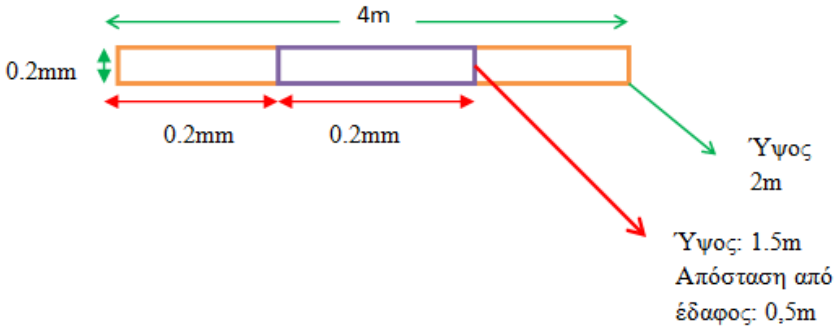
Εικόνα 104 – Κάτοψη ενός τοίχου με συγκεκριμένες διαστάσεις

Παρατηρούμε ότι δεν διαθέτει οπές ούτε έχει συνδεδεμένους άλλους τοίχους επάνω του. Η μετάφρασή του σε τρισδιάστατη γεωμετρία θα γινόταν με το αντικείμενο box (Εικόνα 111).



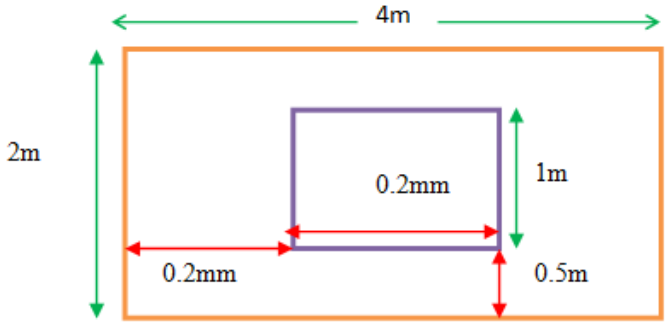
Εικόνα 105 – Τρισδιάστατη αναπαράσταση ενός τοίχου με χρήση box element.

Έστω ότι ο τοίχος περιείχε μια οπή για παράθυρο π.χ. όπως στο παρακάτω σχήμα:



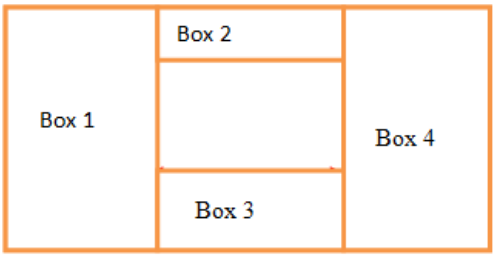
Εικόνα 106 – Κάτοψη ενός τοίχου που περιέχει οπή

Η πλευρική δισδιάστατη απεικόνιση του τοίχου θα ήταν η εξής:



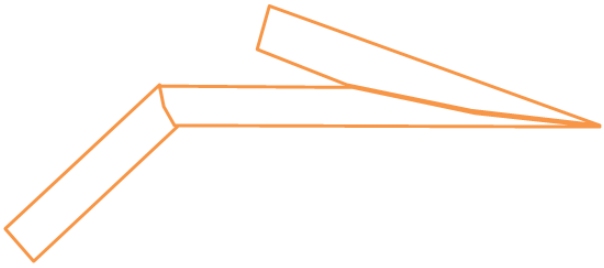
Εικόνα 107 – Πρόσοψη ενός τοίχου που περιέχει οπή

Επειδή η X3D δεν περιλαμβάνει δυνατότητες Boolean μετασχηματισμών σε αντικείμενα ώστε να δημιουργηθούν π.χ. οπές σε αντικείμενα box, ο μόνος τρόπος για να κατασκευαστεί η γεωμετρία του τοίχου, ο οποίος περιέχει οπή, είναι να δομηθεί σε τμήματα χρησιμοποιώντας διαφορετικά στοιχεία box το ένα δίπλα στο άλλο όπως δείχνει το σχεδιάγραμμα (εικόνα 108):



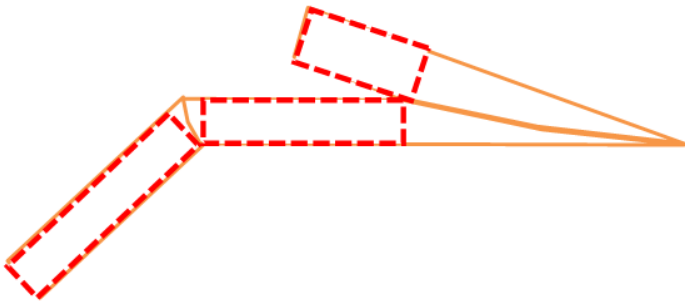
Εικόνα 108 – Τρισδιάστατη αναπαράσταση τοίχου που περιέχει οπή με χρήση πολλαπλών box elements

Έτσι λοιπόν τοποθετούνται στην σειρά boxes και όταν πλησιάζει η γεωμετρία κοντά σε οπή, ανάλογα με τις διαστάσεις της, τοποθετούνται δύο στοιχεία box πάνω και κάτω με κενό ανάμεσά τους και μήκος ανάλογο με το μήκος της οπής (boxes 2,3). Κατόπιν τοποθετούνται στοιχεία box με ύψος από το δάπεδο μέχρι το ύψος του τοίχου, μέχρι να συναντηθεί άλλη οπή. Αυτή η διαδικασία υλοποιείται στην συνάρτησης exportWalls(). Υπάρχει περίπτωση ο τοίχος να είναι συνδεδεμένος με άλλους τοίχους στις άκρες του όπως δείχνει το σχήμα (εικόνα 115):



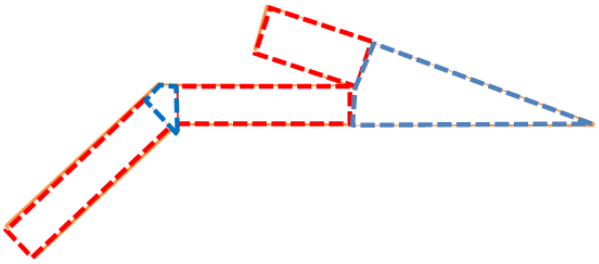
Εικόνα 109 – Κάτοψη τοίχου που περιλαμβάνει συνδέσεις με άλλους τοίχους

Τότε η κατασκευή του, σε τρισδιάστατο χώρο, γίνεται πάλι με στοιχεία box, αλλά θεωρείται ότι ο τοίχος είναι ορθογώνιος και ξεκινάει από τα σημεία που παρουσιάζονται στο σχήμα:



Εικόνα 110 – Τοποθέτηση box elements για την τρισδιάστατη αναπαράσταση τοίχων σε σύνδεση

Η συνάρτηση exportWalls() χτίζει την γεωμετρία των τοίχων χρησιμοποιώντας boxes στις διακεκομμένες με κόκκινο ορθογώνιες περιοχές του σχήματος. Τα υπόλοιπα κενά συμπληρώνονται δημιουργώντας κατάλληλα πολύγωνα για τους κόμβους σύνδεσης, τα οποία αποκτούναι βάθος χρησιμοποιώντας μια τεχνική που ονομάζεται extrusion. Ένα extrusion στην ουσία είναι η δημιουργία ενός τρισδιάστατου αντικειμένου από μια δισδιάστατη επιφάνεια, στην οποία έχει δοθεί βάθος προς τον τρίτο άξονα. Στο συγκεκριμένο παράδειγμα, τα extrusions σημειώνονται με μπλε διακεκομμένο χρώμα.



Εικόνα 111 – Τοποθέτηση extrusions ανάμεσα από τα box elements για την τρισδιάστατη αναπαράσταση των κόμβων σύνδεσης

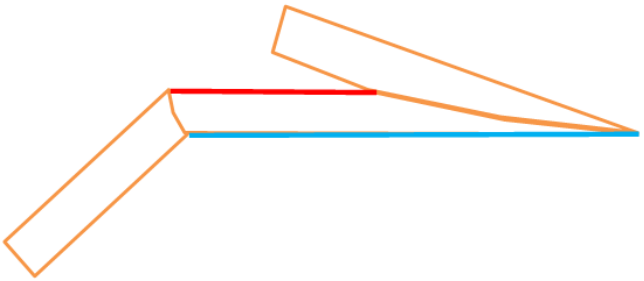
Για να περιγραφεί η διαδικασία του extrusion σε γλώσσα X3D, χρειάζεται ένα πολύγωνο που περιγράφει την επιφάνεια και μια κατεύθυνση στην οποία θα δοθεί βάθος. Η κατεύθυνση ορίζεται να είναι στον άξονα του ύψους, που σε γλώσσα X3D είναι ο άξονας y και όχι ο z.

Το πολύγωνο δίνεται από την ιδιότητα του handle στο συγκεκριμένο κόμβο fillArray η οποία περιέχει τα points του πολυγώνου στην περιοχή σύνδεσης τα οποία υπολογίζονται κάθε φορά που καλείται η μέθοδος calibrateWalls() την οποία περιγράψαμε σε προηγούμενο σημείο της διατριβής.

Η συνάρτηση που δημιουργεί τα extrusions, στις περιοχές σύνδεσης των τοίχων, είναι η exportFills(). Δέχεται ένα handle σαν όρισμα _h, από το οποίο εξετάζει το fillArray (_h.fillArray) διαβάζοντας τα points που είναι αποθηκευμένα. Με βάση εκείνα δημιουργεί το στοιχείο extrusion.

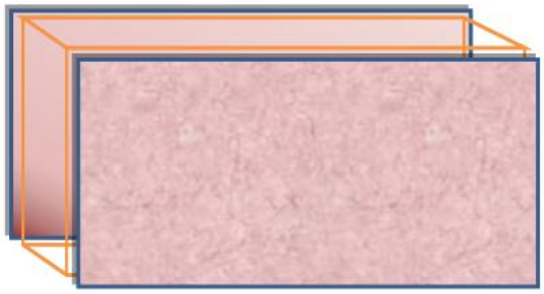
5.7.2 Εξαγωγή διακόσμησης τοίχων σε X3D

Όπως αναφέρθηκε, οι τοίχοι αποτελούνται από πολλά X3D στοιχεία τύπου box τοποθετημένα το ένα δίπλα στο άλλο. Δυστυχώς η X3D δεν μας δίνει έναν τρόπο να εφαρμόσουμε μια ενιαία υφή στο σύνολο αυτών των boxes ώστε να υλοποιήσουμε μια ταπετσαρία στον τοίχο. Ακόμα όμως και αν είχαμε, την απλή περίπτωση του παραλληλόγραμμου τοίχου, στον οποίο δεν συνδέονται άλλοι τοίχοι στα πλευρά του, αλλά ούτε υπάρχουν οπές, πάλι είναι αδύνατον να προσδώσουμε 2 διαφορετικές υφές στις πλευρές του, ή να προσδώσουμε δύο διαφορετικά χρώματα. Ο τρόπος που ξεπεράστηκε αυτός ο περιορισμός ήταν η δημιουργία επιπλέον γεωμετρίας. Πρόκειται για γεωμετρία επιφάνειας η οποία τοποθετείται σε μια μικρή σχετική απόσταση πάνω από τις πλευρές των τοίχων που θέλουμε να διακοσμήσουμε. Για έναν τοίχο ο οποίος είναι συνδεδεμένος με άλλους παρατηρούμε ότι υπάρχει διαφορά στο μήκος και την τοποθέτηση των πλευρών όπως δείχνει το σχήμα:



Εικόνα 112 – Κάτοψη με επισήμανση των περιοχών διακόσμησης ενός τοίχου.

Αυτό που παρατηρούμε στην παραπάνω σχηματική κάτοψη με μπλε και κόκκινο χρώμα, είναι δύο επιφάνειες οι οποίες σκεπάζουν τις πλευρές του τοίχου και δημιουργούν την ψευδαίσθηση ότι έχει διακοσμηθεί με ταπετσαρία. Παρατηρούμε την εφαρμογή αυτών τον επιφανειών σε έναν απλό παραλληλόγραμμο τοίχο:



Εικόνα 113 – Εφαρμογή επιφανειών διακόσμησης σε τρισδιάστατη αναπαράσταση τοίχου.

Τι γίνεται όμως όταν στον τοίχο υπάρχουν οπές? Τότε με κατάλληλα μέσα θα πρέπει να δημιουργηθούν αντίστοιχες οπές στις επιφάνειες εκείνες, οι οποίες συμπίπτουν κατά την τοποθέτηση με τις οπές του τοίχου. Παρακάτω σχηματικά, παρουσιάζεται η γεωμετρία μιας επιφάνειας με οπές, έτοιμη για να τοποθετηθεί σε έναν τοίχο:

Εικόνα 114 – Δημιουργία γεωμετρίας περιοχών διακόσμησης τοίχου που περιλαμβάνει οπή.

Η παραπάνω γεωμετρία δημιουργείται με low level στοιχεία όπως vertices, edges , faces και polygons. Στο παραπάνω σχήμα (Εικόνα 114), με κόκκινες βούλες επισημαίνονται τα σημεία των vertices, με πράσινες γραμμές οι ακμές edges που τα ενώνουν και με γαλάζιο γέμισμα το εσωτερικό των faces που δημιουργούνται.

Η παραπάνω γεωμετρία, μπορεί να δημιουργηθεί με το στοιχείο X3D indexed faceset, το οποίο στην ουσία επιτρέπει την δήλωση μιας λίστας σημείων (vertices) με τις συντεταγμένες τους και ένα διακριτικό index στο καθένα και κατόπιν μιας λίστας ακμών η οποία περιλαμβάνει κύκλους από indexes των vertices για να δηλώσει της πλευρές τις οποίες δημιουργούν. Δυστυχώς δεν υπήρχαν βοηθητικές συναρτήσεις και μέθοδοι για να διαχειριστούμε τέτοιες πρωτογενείς μορφές γεωμετρίας οπότε υλοποιήθηκαν μια σειρά από κλάσεις για την διαχείριση των vertices, των edges, των faces και των πολυγώνων εν τέλει που δημιουργούν.

Ξεκινάμε από το vertex το οποίο στην ουσία είναι ένα σημείο στον χώρο και υλοποιήθηκες σε γλώσσα Javascript από την κλάση vertex η οποία περιγράφεται στο αρχείο export.js:

ΚΩΔΙΚΑΣ #37:

```
1. function Vertex(x,y,z,i)
2. {
3. this.x = x;
4. this.y = y;
5. this.z = z;
6. this.index = i;
7. }
```

Παρατηρούμε όπως είναι λογικό ότι έχει την τριάδα των properties x,y,z για να περιγράφουν τις συντεταγμένες του σημείου στον χώρο καθώς και το διακριτικό index.

Ακολουθεί η συνάρτηση face η οποία περιλαμβάνει μια λίστα από τα vertices (τοποθετημένα σε σειρά) που την δημιουργούν. Η λίστα υλοποιείται με μια απλή διάταξη/array:

ΚΩΔΙΚΑΣ #38:

```
1. function Face()
2. {
3. this.vertices = new Array();
4. }
```

Τα στοιχεία της X3D indexedFaceset υλοποιούνται ως αντικείμενα από την κλάση indexedFaceset η οποία δηλώνεται παρακάτω:

ΚΩΔΙΚΑΣ #39:

```
1. function IndexedFaceset()
2. {
3. this.faces = new Array();
4. this.vertices = new Array();
5. }
```

Η κλάση περιλαμβάνει δύο λίστες, μία για τα vertices και μια για τα faces που δημιουργούνται. Επίσης περιλαμβάνονται μέθοδοι για την προσθήκη vertices και faces. Όταν τελειώσει η περιγραφή της γεωμετρίας γίνεται export σε X3D η περιγραφή του στοιχείου χρησιμοποιώντας τις συναρτήσεις exportFaces() και exportVertices(). Οι συναρτήσεις createFaceRect() και createFaceHole χρησιμοποιούνται βοηθητικά για να δημιουργήσουνε επιφάνειες με την βοήθεια αντικειμένων IndexedFaceset. Η συνάρτηση που δημιουργεί όλες τις επιφάνειες στον τρισδιάστατο χώρο είναι η exportTapestry()

5.7.3 Δημιουργία γεωμετρίας δαπέδων

Τα δάπεδα έχουν παρόμοιο σχήμα με την οριοθέτηση των δωματίων, δηλαδή πολύγωνο. Πρόκειται για πολυγωνικές επιφάνειες οι οποίες διαθέτουν πάχος. Έτσι λοιπόν επαναχρησιμοποιείται το στοιχείο extrusion της X3D για να δημιουργηθεί μια πολυγωνική επιφάνεια με προέκταση στον άξονα του ύψους ώστε να αποκτήσει βάθος. Το πολύγωνο δίνεται από τον πίνακα των points του αντικειμένου floor.Η διακόσμηση με υφές των δαπέδων και των οροφών γίνεται δημιουργώντας ξεχωριστές επιφάνειες χρησιμοποιώντας στοιχεία indexedFaceset.

Η συνάρτηση που αναλαμβάνει την κατασκευή της τρισδιάστατης γεωμετρίας των δαπέδων είναι η export floors:

ΚΩΔΙΚΑΣ #40:

```
1. function exportFloors(xdoc,f)
2. {

3. // get the polygon points
4. var par = f.points;
5. var dcol = f.color.toX3D();
6. if (par.length == 0)
7. return;
8. var crs = "";
9. var i;
10. for (i in par) {
11. crs = crs + " " + par[i].x + " " + par[i].y + ",";
12. }
13. crs = crs + « « + par[0].x + « « + par[0].y;
14. var yy = par[0].y;
```

```
15. var xx = par[0].x;
16. var spl = "0 0 0 , 0 " + 1 + " 0 ";
17. var xtr1 = xdoc.createElement("Extrusion");
18. xtr1.setAttribute("crossSection", crs);
19. xtr1.setAttribute("spine", spl);
20. xtr1.setAttribute("solid", "FALSE");
21. xtr1.setAttribute("convex", "FALSE");
22. var shape = xdoc.createElement("Shape");
23. var app = xdoc.createElement("Appearance");
24. var mat = xdoc.createElement("Material");
25. mat.setAttribute("diffuseColor", dcol);
26. app.appendChild(mat);
27. shape.appendChild(app);
28. shape.appendChild(xtr1);
29. var wrap = xdoc.createElement("Transform");
30. wrap.appendChild(shape);
31. var scene = xdoc.getElementsByTagName("Scene")[0];
32. scene.appendChild(wrap);
33. }
```

5.7.4 Δημιουργία κάμερας περιήγησης

Κάθε αρχείο X3D απαιτεί περιγραφή της κάμερας από την οποία θα κοιτάζει ο χρήστης την σκηνή. Κάθε αρχείο X3D μπορεί να περιέχει στιγμιότυπα κάμερας τα οποία ο χρήστης μπορεί να εναλλάσσει. Για τον καθορισμό τους χρησιμοποιούνται στοιχεία X3D τύπου <viewpoint>. Απαιτείται τουλάχιστον την κεντρική κάμερα περιήγησης της σκηνής την οποία χειρίζεται ο χρήστης για να προχωράει στους χώρους και τα δωμάτια. Η exportViewPoint() είναι υπεύθυνη για την δημιουργία του στοιχείου X3D που αναπαριστά την κάμερα περιήγησης μέσα στην τρισδιάστατη σκηνή (ύψος, σημείο, κατεύθυνση κτλ). Ο κώδικάς της είναι ο εξής:

ΚΩΔΙΚΑΣ #41:

```
1.  function exportViewPoint(xdoc)
2.  {
3.  var viewpoint = xdoc.createElement("Viewpoint");
4.  viewpoint.setAttribute("DEF","MyviewPoint")
5.  viewpoint.setAttribute("description","default");
6.  viewpoint.setAttribute("position"," 300 80 300 ");
7.  viewpoint.setAttribute("orientation","0 0 1 0 ");
8.  viewpoint.setAttribute("FOV","1.067");
9.  viewpoint.setAttribute("centerOfRotation","300 300 300");
10. viewpoint.setAttribute("jump","TRUE");
11. var nav = xdoc.createElement("NavigationInfo");
12. nav.setAttribute("avatarSize","5 80 10");
13. nav.setAttribute("headlight","TRUE");
14. nav.setAttribute("speed","60");
15. nav.setAttribute("type","WALK");
16. var sun = xdoc.createElement("PointLight")
17. sun.setAttribute("on","TRUE");
18. sun.setAttribute("intensity","1");
19. sun.setAttribute("ambientIntensity","0");
20. sun.setAttribute("color","1 1 1");
21. sun.setAttribute("location","300 200 300");
22. sun.setAttribute("attenuation"," 1 0 0 ");
```

```
23. sun.setAttribute("radius","200");
24. var scene = xdoc.getElementsByTagName("Scene")[0];
25. scene.appendChild(viewpoint);
26. scene.appendChild(nav);
27. scene.appendChild(sun);
28. }
```

Παρατηρούμε ότι εκτός από το viewpoint η συνάρτηση δημιουργεί και ένα αντικείμενο τύπου <NavigationInfo> για την περιγραφή των παραμέτρων του avatar του χρήστη στην σκηνή.

5.7.5. Εισαγωγή τρισδιάστατων αντικειμένων διακόσμησης

Τα τρισδιάστατα αντικείμενα διακόσμησης δημιουργούνται κάνοντας εισαγωγή γεωμετρίες που περιγράφονται σε εξωτερικά αρχεία. Τα αρχεία αυτά περιλαμβάνονται στον κατάλογο assets του διακομιστή, και είναι μορφής X3D. Μαι μπορούν να τοποθετηθούν μέσα στο τρέχων αρχείο χρησιμοποιώντας το στοιχείο <inline>. Το στοιχείο αυτό δέχεται σαν attribute το url του αρχείου που περιγράφει το αντικείμενο. Επίσης εφαρμόζονται κατάλληλα transformations περιστροφής και μετατόπισης ώστε το αντικείμενο να ενσωματωθεί στο ακριβές σημείο, με τον ακριβή προσανατολισμό που όρισε ο χρήστης. Υπεύθυνη για αυτές τις ενέργειες είναι η συνάρτηση exportObjects() η οποία δημιουργεί όλα τα απαραίτητα στοιχεία τύπου <inline>.

5.7.6. Αποθήκευση της σκηνής

Κατά την εκτέλεση των συναρτήσεων δημιουργίας της τρισδιάστατης γεωμετρίας τοποθετούνται κατάλληλα elements, στην ουσία nodes, σε ένα Document Object Model που περιγράφει το X3D αρχείο μας. Η συνάρτηση savetheDay() αναλαμβάνει την αποθήκευσή του, στο δίσκο χρησιμοποιώντας τεχνολογία ajax. Ο κώδικας της συνάρτησης είναι ο εξής:

ΚΩΔΙΚΑΣ #42:

```
1.  function savetheday(filename)
2.  {
3.  var http = new XMLHttpRequest();
4.  var url = "save.PHP";
5.  var str = testXML();
6.  var params = filename + str;
7.  http.open("POST", url, true);
8.  //Send the proper header information along with the request
9.  http.setRequestHeader("Content-type",         "application/x-www-form-
    urlencoded");
10. http.setRequestHeader("Content-length", params.length);
11. http.setRequestHeader("Connection", "close");
12. http.onreadystatechange = function() {//Call a function when the state
    changes.
13. if(http.readyState == 4 && http.status == 200) {
```

```
14. }
15. }
16. http.send(params);}
```

Παρατηρούμε ότι η συνάρτηση δημιουργεί ένα XMLHttpObject και του διοχετεύει τα απαραίτητα στοιχεία με μέθοδο POST. Καθορίζονται και οι requestHeaders και το string που περιγράφει το X3D αρχείο μας στέλνεται στο script save.PHP.

ΚΩΔΙΚΑΣ #43:

```
1. <?PHP
2. $content = $_POST["desX3D"];
3. $filename = $_POST["filename"];
4. $file = fopen($filename,"w+");
5. fwrite($file,$content);
6. fclose($file);
7. echo "Design X3D Scene Saved!";
8. ?>
```

Παρατηρούμε, ότι το παραπάνω script δημιουργεί ένα αρχείο, ανάλογα με το filename που έχει διοχετευτεί από την εφαρμογή και χρησιμοποιεί την fwrite(), για να το αποθηκεύσει στον δίσκο του εξυπηρετητή. Με παρόμοιο τρόπο, λειτουργεί και το σύστημα αποθήκευσης των σχεδίων κατόψεων, τα οποία αποθηκεύονται σαν ένα xml αρχείο. Το αρχείο περιλαμβάνει περιγραφή των δομών δεδομένων που υπεύθυνες για την δημιουργία των αντικειμένων handles, walls, floors, κτλ. Η σειριακή αποθήκευση και φόρτωση των σχεδίων κατόψεων υλοποιείται στο αρχείο xmlio.js

ΚΕΦΑΛΑΙΟ 6
ΑΠΟΤΕΛΕΣΜΑΤΑ ΚΑΙ ΕΠΕΚΤΑΣΕΙΣ

6.1 Εισαγωγή

Το παρόν κεφάλαιο περιλαμβάνει μια σύνοψη των όσων μελετήθηκαν, σχεδιάστηκαν και υλοποιήθηκαν σε όλο το μήκος αυτής της διατριβής. Παρουσιάζονται τα τελικά αποτελέσματα και γίνεται εκτίμηση του κατά πόσο επετεύχθησαν οι αρχικοί στόχοι, καθώς παρουσιάζονται και μελλοντικές επεκτάσεις.

6.2 Στόχος της εφαρμογής

Ο στόχος υλοποίησης της εφαρμογής ήταν να χρησιμοποιηθούν σύγχρονες τεχνολογίες και στάνταρ του παγκόσμιου ιστού και να συνεργαστούν με τέτοιο τρόπο μεταξύ τους, ώστε να δημιουργηθεί μια διαδικτυακή εφαρμογή με δυνατότητες και διαδραστικότητα μιας αντίστοιχης desktop. Δόθηκε βάση στο να χρησιμοποιηθούν όσον τον δυνατόν περισσότερες τεχνολογίες ανοιχτού κώδικα, οι οποίες υποστηρίζονται εγγενώς από τους φυλλομετρητές, δεν απαιτούν την ύπαρξη plugins και ορίζονται από διεθνή κονσόρτσιουμ ως web standards. Αυτό ήταν απαραίτητο ώστε να αποδεσμευτεί η εφαρμογή μας από οποιεσδήποτε εξαρτήσεις οι οποίες θα δυσχεραίνανε το portability της. Στόχος ήταν να εκτελείται, απαιτώντας μόνο από τον χρήστη την ύπαρξη ενός φυλλομετρητή. Επιλέχθηκε σαν πλατφόρμα εκτέλεσης ο φυλλομετρητής Mozilla Firefox, ο οποίος φημίζεται για την μεγάλη και ορθή υποστήριξη των διάφορων web standards. Επιλέχθηκε υποστήριξη μόνο για έναν φυλλομετρητή ώστε να αποφευχθεί η επιπλέον συγγραφή κώδικα για την κάλυψη των ιδιομορφιών άλλων προγραμμάτων περιήγησης στον ιστό.

6.3 Υλοποίηση της εφαρμογής

Όπως αναφέρθηκε και στα προηγούμενα κεφάλαια της διπλωματικής εργασίας η υλοποίηση της εφαρμογής απαιτούσε την χρησιμοποίηση πολλών διαφορετικών τεχνολογιών. Καταρχήν η ανάπτυξη και η εκτέλεση έγινε σε περιβάλλον windows 7. Για το κομμάτι του διακομιστή web χρησιμοποιήθηκε το πακέτο αυτόματης εγκατάστασης της τριάδας (apache,PHP,MySQL) Wamp server 2.0. Για την διαχείριση της βάσης δεδομένων χρησιμοποιήθηκε το command line tool Mysqld

καθώς επίσης και τα MySql gui tools: MySql Query browser για την συγγραφή και εκτέλεση sql queries, και MySql Administrator για την γενική διαχείριση της βάσης και την δημιουργία πινάκων. Για την συγγραφή κώδικα έγινε χρήση του open-source επεξεργαστή κειμένου notepad++, ο οποίος παρέχει syntax highlighting. Ο notepad++ χρησιμοποιήθηκε τόσο για συγγραφή html σελίδων όσο και για αρχεία css, Javascript, PHP, xml, X3D, SVG. Για την δημιουργία bitmap γραφικών για τις ιστοσελίδες χρησιμοποιήθηκε το δωρεάν και ανοιχτού κώδικα πρόγραμμα gimp. Για την δημιουργία vector γραφικών και την εξαγωγή τους σε SVG χρησιμοποιήθηκε το πρόγραμμα ανοικτού κώδικα Inkscape. Για την δημιουργία τρισδιάστατων γραφικών και την εξαγωγή τους σε X3D χρησιμοποιήθηκε το open source πρόγραμμα blender. Η επαλήθευση κατά την κατασκευή των html σελίδων έγινε μέσα από τον browser Firefox, χρησιμοποιώντας την επέκταση web developer extension η οποία επιτρέπει εποπτεία των διάφορων τμημάτων μιας ιστοσελίδας. Η αποσφαλμάτωση του κώδικα Javascript γίνεται με το εργαλείο firebug που αποτελεί extension του Firefox. Η αποσφαλμάτωση του κώδικα PHP έγινε μέσω του φυλλομετρητή, εκτελώντας τοπικά την εφαρμογή και ελέγχοντας τα μηνύματα λαθών του server.

6.4 Αποτελέσματα

Η εφαρμογή από τα πρώτα λειτουργικά στάδια της ανάπτυξής της πέρασε από έναν κύκλο επαναλήψεων σε χρήση και αλλαγές. Για την χρήση της εφαρμογής επιλέχθηκε μια ομάδα χρηστών από τους οποίους άλλοι είχαν εμπειρία στο διαδίκτυο, άλλοι όχι και άλλοι είχανε εμπειρία στην χρήση σχεδιαστικών προγραμμάτων cad.

Κατά την χρήση της εφαρμογής σημειώθηκαν αρκετές αρχικές ατέλειες στα εργαλεία σχεδίασης. Ιδιαίτερα εποικοδομητικές ήταν οι παρατηρήσεις των χρηστών που είχαν εμπειρία σε εφαρμογές αρχιτεκτονικής σχεδίασης. Ένα από τα πρώτα πράγματα που παρατηρήθηκε ήταν ότι η σχεδίαση των χώρων με βάση τα δωμάτια από τα οποία αποτελούνται ήταν περιοριστικό μοντέλο. Έτσι επικεντρωθήκαμε στην ανάπτυξη των εργαλείων σχεδίασης τοίχων καθώς αυτοί αποτελούν τον σκελετό κάθε σχεδίου κατόψεως. Παρατηρήθηκε ότι οι χρήστες είχανε μεγαλύτερο έλεγχο στην δομή των χώρων χρησιμοποιώντας την ευέλικτη δομή σύνδεσης των τοίχων.

Όσον αφορά την διακόσμηση των δομικών αντικειμένων (τοίχοι, δάπεδα οροφές κτλ), αρχικά οι επιλογές του χρήστη στις υφές και στα χρώματα δεν αντανακλώνταν οπτικά στο σχέδιο. Αναφερόντουσαν μόνο στην πλευρική μπάρα σχεδίασης για ενημερωτικούς λόγους. Κατά την χρήση της εφαρμογής παρατηρήθηκε, όσον αφορά την απόδοση υφών και χρωμάτων, ότι πολλοί χρήστες αδυνατούσαν να συνδέσουν το τελικό αποτέλεσμα της τρισδιάστατης απεικόνισης με της επιλογές τους κατά τον σχεδιασμό της κάτοψης. Έτσι δημιουργήθηκαν κατάλληλες δομές και διαδικασίες οι οποίες επέτρεψαν να δημιουργηθούν SVG patterns και fills ώστε να φαίνονται οπτικά στο δισδιάστατο σχέδιο οι υφές που χρησιμοποιούνται.

Μια άλλη παρατήρηση που προέκυψε από την χρήστη της εφαρμογής ήταν η ανυπαρξία εργαλείων σχεδίασης με ακρίβεια. Π.χ. δυνατότητα σχεδίασης ορθογώνιων τοίχων και η δυνατότητα ύπαρξης grid στο πίσω μέρος της εφαρμογής ως οδηγός. Αυτό είχε ως αποτέλεσμα την δυσκολία στο να δημιουργηθούν ορθές γωνίες στα δωμάτια αλλά και να γίνει σχεδίαση των χώρων με συγκεκριμένες διαστάσεις. Έτσι υλοποιήθηκε το σύστημα των οδηγών το οποίο ενημερώνει τον χρήστη για τις ακριβείς διαστάσεις των αντικειμένων. Επίσης δημιουργήθηκε και το σύστημα πλέγματος στο φόντο της σχεδίασης καθώς επίσης και η δυνατότητα επιβολής περιορισμών κατά την κίνηση του ποντικιού (κρατώντας το πλήκτρο shift).

Επίσης αρκετοί από τους χρήστες παρατήρησαν στα πρώτα στάδια της εφαρμογής ότι ήταν εμφανής η έλλειψη υποστήριξης διαφορετικών μονάδων μέτρησης. Αυτό διορθώθηκε με την προσθήκη ειδικών κλάσεων και διαδικασιών μετατροπής των μονάδων μέτρησης.

Πάρα πολύ σημαντικό εργαλείο αποτέλεσε και το firebug το οποίο εκτός από debugging διαθέτει και profiling tools. Τα εργαλεία αυτά χρησιμοποιήθηκαν κατά κόρων κατά την ανάπτυξη για να συλλέξουμε πληροφορίες σχετικά με την ταχύτητα εκτέλεσης της Javascript και την κατανάλωση πόρων. Έτσι κατάφεραν να εντοπιστούν κομμάτια του κώδικα που δημιουργούσαν καθυστέρηση στο σύστημα και έγινε optimization.

6.5 Μελλοντικές προεκτάσεις και βελτιώσεις της εφαρμογής

Στην παράγραφο αυτή θα αναφερθούν πιθανές μελλοντικές επεκτάσεις και βελτιώσεις της εφαρμογής. Πολλές από τις παρακάτω ιδέες που εξετάζονται προέκυψαν από την χρήση της εφαρμογής

- **Δημιουργία συστήματος βοήθειας για τον χρήστη.**
  Μπορεί να γίνει προσθήκη ενός συστήματος βοήθειας με απλά tutorials και animations τα οποία να εξηγούν στον άπειρο χρήστη τα εργαλεία σχεδίασης της εφαρμογής καθώς και πως μπορούν να επιτευχθούν διάφορες λειτουργίες.

- **Σχεδίαση με την βοήθεια οδηγών.**
  Αρκετές σχεδιαστικές διαδικασίες θα μπορούσαν να αυτοματοποιηθούν με την βοήθεια wizards. Οι wizards αυτοί μέσα από απλά βήματα θα ζητούσαν από τον χρήστη συγκεκριμένες πληροφορίες και παραμέτρους για τους χώρους που επιθυμεί να σχεδιάσει. Κατόπιν σύμφωνα με τις επιλογές του θα παρήγαγαν αυτόματα τα δωμάτια και τα αντικείμενα που τα διακοσμούν.

- **Εφαρμογή αρχιτεκτονικών κανόνων.**
  Θα μπορούσαν να προστεθούν διαδικασίες οι οποίες να κάνουν ελέγχους στις κατόψεις του χρήστη για αρχιτεκτονικά λάθη και να τον ενημερώνουν ανάλογα. Επίσης θα μπορούσαν να προστεθούν διαδικασίες object collision

που να μην επιτρέπουν την τοποθέτηση αντικειμένων οπουδήποτε στο σχέδιο (όπως μέσα από τοίχους κτλ).

- **Εξαγωγή σε ανοιχτά formats εικόνας και 3d.**
  Μπορεί να προστεθούν διαδικασίες που επιτρέπουν την εξαγωγή του σχεδίου σε διάφορα formats εικόνας. Επίσης θα μπορούσαν να δημιουργηθούν διαδικασίες για την εξαγωγή του τρισδιάστατου σχεδίου σε open-source formats όπως το collada.

- **Χρησιμοποίηση Webgl ή O3D.**
  Λόγω της χρησιμοποίησης της τεχνολογίας X3D είναι απαραίτητη η ύπαρξη κάποιου plugin για την αναπαράσταση των τρισδιάστατων σκηνών. Μπορούν μελλοντικά, αφού τα Webgl και X3Dom αποτελέσουν πλήρως υποστηριγμένα standards, να δημιουργηθούν κατάλληλες διαδικασίες σε κώδικα Webgl ή και O3D για την αναπαράσταση των σκηνών και την περιήγηση σε αυτές χωρίς χρήση plugin.

- **Δημιουργία administrative σελίδας για τον έλεγχο του παρασκηνίου της εφαρμογής.**
  Επειδή η διαχείριση της εφαρμογής απαιτεί την επισκόπηση διάφορων παραμέτρων, διάφορων τεχνολογιών θα μπορούσε να δημιουργηθεί ένα online τμήμα της εφαρμογής το οποίο να επιτρέπει την διαχείριση του παρασκηνιακού συστήματος (λογαριασμοί χρηστών, βάσεις δεδομένων αντικειμένων, υφών κτλ, έλεγχος του file system του διακομιστή) μέσω ιστοσελίδας χωρίς να χρειάζονται ftp εργαλεία επικοινωνίας με τον server ή command line scripts για την διαχείριση της βάσης.

- **Περισσότερα εργαλεία σχεδίασης.**
  Μπορούν να προστεθούν ακόμα περισσότερα εργαλεία σχεδίασης ώστε να μπορεί ο χρήστης να έχει μεγαλύτερο έλεγχο στον τρόπο που δημιουργεί τα σχέδια των κατόψεων. Θα μπορούσαν επίσης να τοποθετηθούν εργαλεία σχεδίασης αντικειμένων, όπως πάγκοι, ράφια, ντουλάπες, εντοιχίσεις κτλ.

- **Περισσότερα αντικείμενα.**
  Από τις ποιο απλές επεκτάσεις της εφαρμογής είναι η δημιουργία και προσθήκη περισσότερων αντικειμένων, τόσο υφών όσο και τύπων παραθύρων, πορτών, φωτιστικών, επίπλων, ηλ. Συσκευών κτλ. Θα μπορούσε επίσης να δημιουργηθεί και ένα εργαλείο το οποίο να επιτρέπει upload αντικειμένων στην βάση της εφαρμογής.

6.6 Επίλογος

Υλοποιώντας την παραπάνω εργασία, στάθηκα αρκετά τυχερός, διότι μου δόθηκε η ευκαιρία να ασχοληθώ σε βάθος με τις σύγχρονες τεχνολογίες ανάπτυξης εφαρμογών στον παγκόσμιο ιστό. Η προσπάθεια δημιουργίας μιας εφαρμογής σχεδιασμού δισδιάστατου και τρισδιάστατου περιεχομένου ήδη αποτελεί πρόκληση στον desktop τομέα. Ακόμα μεγαλύτερη πρόκληση αποτέλεσε η προσπάθεια υλοποίησής της μέσα στα πλαίσια ενός φυλλομετρητή, με όλους τους περιορισμούς που παρουσιάζει σαν περιβάλλον ανάπτυξης. Αυτό είχε σαν αποτέλεσμα να μελετηθούν αρκετές διαφορετικές τεχνολογίες για την υπέρβαση αυτών των περιορισμών. Μεγάλη πρόκληση αποτέλεσε επίσης ο αρμονικός συνδυασμός των τεχνολογιών αυτών και η μεταξύ τους επικοινωνία. Λήφθηκαν υπόψη οι μικρότερες ταχύτητες εκτέλεσης των δυναμικών γλωσσών καθώς και οι καθυστερήσεις απόκρισης με τον διακομιστή.

Η υλοποίηση του editor σχεδιασμού κατόψεων απαίτησε μελέτη και κοπιαστική δουλειά ώστε να δημιουργηθεί ένα κατάλληλο σύστημα δομών υποστήριξης των δισδιάστατων σχημάτων που παρουσιάζονται στην οθόνη. Δόθηκε η ευκαιρία να ασχοληθούμε με αντικείμενα κατασκευής τρισδιάστατης γεωμετρίας σε επίπεδο vertices και faces.

Η τάση εμφάνισης όλων και περισσότερων web applications στο διαδίκτυο είναι σημάδι μιας νέας εποχής που πλησιάζει. Χαρακτηριστικό της είναι η μεταφορά των desktop εφαρμογών που είχαμε συνηθίσει να εγκαθιστούμε στο λειτουργικό μας τοπικά, να μεταφέρονται στον ιστό με διαδικτυακή μορφή μέσα από ιστοσελίδες υψηλής διαδραστικότητας. Η επεξεργασία και αποθήκευση του περιεχομένου θα γίνεται πάντα στο «σύννεφο» του ιστού κάτι που θα επιτρέπει την πρόσβαση σε αυτό από οποιοδήποτε μέρος, οποιοδήποτε υπολογιστή, οποιοδήποτε λειτουργικό.

Βιβλιογραφία

Achour, M., Betz, F., Dovgal, A., Lopes, N., Magnusson, H., Richter, G., και συν. (2010). *PHP Manual.* Ανάκτηση από PHP.net : http://www.PHP.net/manual/en/

adobe. (n.d.). *Adobe Flash Platform.* Ανάκτηση από www.adobe.com: http://www.adobe.com/flashplatform/

Allsopp, J. (2009). *Developing with Web Standards.* New Riders Press.

Asleson, R., & Schutta, N. T. (2005). *Foundations of Ajax.* Apress.

Brinzarea, B., & Darie, C. (2010). *AJAX and PHP: Building Modern Web Applications 2nd Edition.* Packt Publishing.

Brutzman, D., & Daly, L. (n.d.). *X3D: Extensible 3D Graphics for Web Authors.*

Cagle, K. (2002). *SVG Programming: The Graphical Web.* Apress.

Campesato, O. (2003). *Fundamentals of SVG Programming: Concepts to Source Code.* Charles River Media.

Chaffee, A. (2000). *"What is a web application (or "webapp")?".* Ανάκτηση από jguru.com: http://www.jguru.com/faq/view.jsp?EID=129328

Consortium, W. (2008). *Extensible 3D (X3D - Part 1: Architecture and base components - ISO/IEC 19775-1:2008.* Ανάκτηση από Web3dOrg: http://www.web3d.org/X3D/specifications/ISO-IEC-19775-1.2-X3D-AbstractSpecification/index.html

consortium, w. (n.d.). *The Virtual Reality Modeling Language 1.0 Specification.* Ανάκτηση από www.web3d.org: http://www.web3d.org/X3D/specifications/vrml/VRML1.0/index.html

Converse, T., Park, J., & Morgan, C. (2004). *PHP5 and MySQL Bible.* Wiley.

Crockford, D. (2008). *Javascript: The Good Parts.* Yahoo Press.

Crockford, D. (n.d.). *Javascript: The World's Most Misunderstood Programming Language.* Ανάκτηση από www.crockford.com: http://www.crockford.com/Javascript/Javascript.html

Dailey, D. (n.d.). *SVG animation with Javascript and SMIL*. Ανάκτηση από Slipery Rock University : http://srufaculty.sru.edu/david.dailey/SVG/

Daly, L. (2002). *SVG Essentials.* O'Reilly Media.

Dunn, F. (2002). *3D Math Primer for Graphics and Game Development.* ones & Bartlett Publishers.

Evjen, B., Sharkey, K., Thangarathinam, T., Kay, M., Vernet, A., & Ferguson, S. (2008). *Professional XML.* Wrox.

Flanagan, D. (2006). *Javascript: The Definitive Guide.* O'Reilly Media.

Foundation, M. (n.d.). Ανάκτηση από Mozilla Developer Network: ttps://developer.mozilla.org/en-US/

Garrett, J. J. (n.d.). *Ajax: A New Approach to Web Applications.* Ανάκτηση από Adaptive Path: http://www.adaptivepath.com/ideas/essays/archives/000385.PHP

Gay, J. (n.d.). *The History of Flash.* Ανάκτηση από www.adobe.com: http://www.adobe.com/macromedia/events/john_gay/page02.html

Geroimenko, V., & Chen, C. (2005). *Visualizing Information Using SVG and X3D.* Springer.

Gilmore, W. (2002). *Beginning PHP and MySQL 5: From Novice to Professional, Second Edition.* Apress.

Google. (n.d.). Ανάκτηση από Chrome Experiments : http://www.chromeexperiments.com/

Google. (n.d.). Ανάκτηση από Google O3D: http://code.google.com/intl/el-GR/apis/o3d/

Group, K. (n.d.). *WebGL - OpenGL ES 2.0 for the Web*. Ανάκτηση από www.khronos.org: http://www.khronos.org/webgl/

III, A. T. (2008). *Ajax: The Definitive Guide.* O'Reilly Media.

inc., A. (2008). *Dynamic HTML and XML: The XMLHttpRequest Object.* Ανάκτηση από developer.apple.com: http://developer.apple.com/internet/webcontent/xmlhttpreq.html

Institute, F. (n.d.). Ανάκτηση από X3D DOM: http://www.X3Dom.org/

interaktonline. (2005). *Why use AJAX?* Ανάκτηση από www.interaktonline.org: http://www.interaktonline.com/support/articles/Details/Ajax:+Asynchronously+Moving+Forward-Why+use+Ajax%3F.html?id_art=36&id_asc=309

Lengyel, E. (2003). *Mathematics for 3D Game Programming and Computer Graphics, Second Edition.* Charles River Media.

Lerdorf, R., & Tatroe, K. (2002). *Programming PHP*. O'Reilly Media.

Lindsey, K. (n.d.). Ανάκτηση από Kevlin Lindsey Software Development: http://www.kevlindev.com/tutorials/basics/index.htm

Microsoft. (n.d.). Ανάκτηση από Microsoft Silverlight: http://www.silverlight.net/

microsystems, s. (n.d.). Ανάκτηση από JAVA.com: http://java.com/en/

Oracle. (2010). *MySQL 5.0 Reference Manual.* Ανάκτηση από Mysql : http://dev.mysql.com/doc/refman/5.0/en/index.html

Powers, S. (2008). *Learning Javascript, 2nd Edition.* O'Reilly Media.

Ray, E. T. (2002). *Learning XML, Second Edition.* O'Reilly Media.

Sklar, D. (2004). *Learning PHP 5.* O'Reilly Media.

Stefanov, S. (2008). *Object-Oriented Javascript: Create scalable, reusable high-quality Javascript applications and libraries.* Packt Publishing.

Swartz, A. (n.d.). *http://www.aaronsw.com/weblog/ajaxhistory.* Ανάκτηση από Aaron Swartz: http://www.aaronsw.com/weblog/ajaxhistory

Tahaghogh, S. M., & Williams, H. (2006). *Learning MySQL.* O'Reilly Media.

team, a. (n.d.). Ανάκτηση από Away3d flash engine: http://away3d.com/

team, p. (n.d.). Ανάκτηση από papervision3d: http://blog.papervision3d.org/

W3C. (2002). *XHTML™ 1.0 The Extensible HyperText Markup Language.* Ανάκτηση από World Wide Web Consortium: http://www.w3.org/TR/xhtml1/

W3C. (2008). *Extensible Markup Language (XML) 1.0 (Fifth Edition) Specification.* Ανάκτηση από W3C: http://www.w3.org/TR/REC-xml/

W3C. (2010). *Scalable Vector Graphics (SVG) 1.1 (Second Edition) Specification.* Ανάκτηση από W3C: http://www.w3.org/TR/SVG/

Watt, A. H., & Lilley, C. (2002). *SVG Unleashed.* Sams.

Welling, L., & Thomson, L. (2008). *PHP and MySQL Web Development (4th Edition).* Addison-Wesley Professional.

yahoo. (n.d.). *SVG developers.* Ανάκτηση από yahoo groups: http://tech.groups.yahoo.com/group/SVG-developers/

Zakas, N. C. (2009). *Professional Javascript for Web Developers.* Wrox.

Zakas, N. C., McPeak, J., & Fawcett, J. (2007). *Professional Ajax, 2nd Edition.* Wrox.