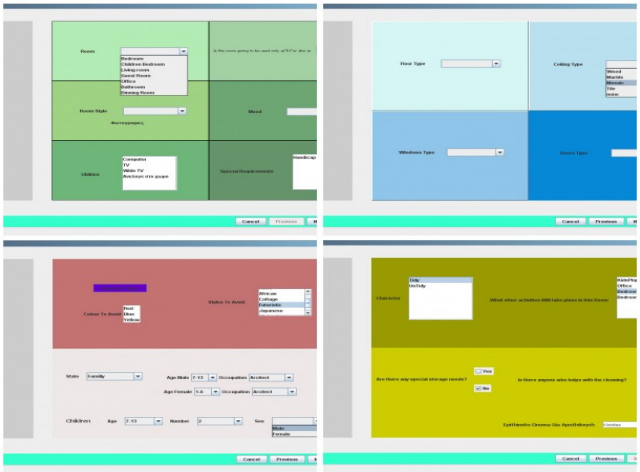
Τ Ε Χ Ν Ο Λ Ο Γ Ι Κ Ο Ε Κ Π Α Ι Δ Ε Υ Τ Ι Κ Ο

Ιδρυμα Κρητης

Τμημα Εφαρμοσμενης Πληροφορικης & Πολυμεσων

Π Τ Υ Χ Ι Α Κ Η Ε Ρ Γ Α Σ Ι Α

Δημιουργία συστήματος επιλογής καλύτερης λύσης σε εφαρμογή σχεδιασμού εσωτερικού χώρου βασισμένου σε παραδείγματα(Case Based Reasoning)



ΚΑΡΠΟΥΖΑΣ ΚΩΣΤΑΣ -1294-

Επιβλεπων Καθηγητης:

Μαλάμος Αθανάσιος

ΗΡΑΚΛΕΙΟ 2008

ΠΕΡΙΕΧΟΜΕΝΑ

1 ΚΕΦΑΛΑΙΟ 4

1.1 Σκοπός Πτυχιακής Εργασίας 4

1.2 Case-based reasoning 4

1.3 Xml 5

1.4 Xpath 6

1.5 Xquery 6

2 ΚΕΦΑΛΑΙΟ 8

Λειτουργικότητα 8

3 ΚΕΦΑΛΑΙΟ 20

3.1 Ανάλυση της Κλάσης Αρχική.java 20

3.1.1 Ο κώδικας του ActionListener του jMenuItem1: 20

3.1.2 Ο κώδικας του ActionListener jMenuItem2: 20

3.2 Ανάλυση της κλάσης LoadMyXml.java 21

3.2.1 Επεξήγηση της μεθόδου filljList() 21

3.2.2 Επεξήση του ActionListener του κουμπιού jButton1 22

3.3 Ανάλυση της Κλάσης MyWizard 23

3.3.1 Επεξήγηση της μεθόδου doFill() 26

3.3.2 Επεξήγηση του ActionListener του κουμπιού jButton3 (κουμπί Next) 29

3.3.3 Επεξήγηση του ActionListener του κουμπιού jButton2 (κουμπί Previous) 30

3.3.4 Επεξήγηση της μεθόδου MouseClicked της jLabel9 31

3.3.5 Επεξήγηση του ActionListener του κουμπιού Button4 (κουμπί Finish) 32

3.4 Ανάλυση της κλάσης MiniBrowser 54

3.4.1 Επεξήγηση της μεθόδου doStyle() 54

3.4.2 Επεξήγηση της μεθόδου LoadPage() 56

3.4.3 Επεξήγηση της μεθόδου showPage() 56

3.5 Ανάλυση της Κλάσης Update.java 57

3.5.1 Επεξήγηση της μεθόδου UpdateForm() 57

3.5.2 Επεξήγηση της μεθόδου jButton4ActionPerformed ()(κουμπί Update) 59

3.6 Ανάλυση της κλάσης Proxy.java 62

3.6.1 Επεξήγηση της μεθόδου doExit() 62

3.7 Class Diagram 65

3.7.1 Class Diagram της κλάσης LoadMyXml.java 65

3.7.2 Class Diagram της κλάσης MiniBrowser.java 65

3.7.3 Class Diagram της κλάσης Arxikh.java 66

3.7.4 Class Diagram της κλάσης Proxy.java 67

4 ΒΙΒΛΙΟΓΡΑΦΙΑ 68

1 ΚΕΦΑΛΑΙΟ

1.1 Σκοπός Πτυχιακής Εργασίας

Στην εργασία αυτήν θα υλοποιήσουμε έναν οδηγό μέσω του οποίου ο χρήστης θα περιγράφει τις επιθυμίες του για τον σχεδιασμό χώρων μιας κατοικίας. Μέσα από τον wizard θα επιλεγούμε για παράδειγμα τα χρώματα που επιθυμούμε αλλά και τα χρώματα που θέλουμε να αποφύγουμε για τους συγκεκριμένους χώρους.

Επίσης ο wizard θα παρέχει τη δυνατότητα παρουσίασης μέσω φωτογραφιών δειγμάτων διαφόρων χώρων σπιτιών υλοποιώντας μια δομή query by example.

Τα αποτελέσματα που καταγράφουν τις επιθυμίες του χρήστη θα δημιουργήσουν ένα ιεραρχικό XML. Αυτό θα χρησιμοποιηθεί ώστε μέσω XQuery (γλώσσα ανάκτησης δεδομένων) να δημιουργούνται Queries για την εύρεση σχεδίων χώρων που πληρούν τις προδιαγραφές που θέτει ο χρήστης και είναι αποθηκευμένα στην σε βάση δεδομένων.

1.2 Case-based reasoning

Με την ευρεία έννοια, είναι η διαδικασία λύσης προβλημάτων στηριζόμενη σε λύσεις παρόμοιων προβλημάτων του παρελθόντος.Ένας μηχανικός αυτοκινήτων που φτιάχνει ένα αυτοκίνητο ανακαλώντας στη μνήμη του ένα άλλο αυτοκίνητο που παρουσίασε τα ίδια προβλήματα, χρησιμοποιεί το **Case-based reasoning.**Ένας δικηγόρος που υπερασπίζεται ένα συγκεκριμένο αποτέλεσμα σε μια δίκη βασιζόμενος σε δεδικασμένες υποθέσεις ή ένας δικαστής που δημιουργεί μια νομολογία χρησιμοποιούν το **Case-based reasoning.** Παρόμοια ένας μηχανικός που αντιγράφει λειτουργικά στοιχεία από τη φύση (χρησιμοποιώντας τον βιομιμητισμό), χρησιμοποιεί τη φύση σαν μια βάση δεδομένων για τις λύσεις των προβλημάτων του.
Case-based reasoning είναι ένα εξέχον είδος δημιουργίας αναλογιών.
Υποστηρίζεται ότι το **Case-based reasoning** δεν είναι μόνο μια πολύ ισχυρή δύναμη για υποθέσεις υπολογιστών, αλλά και μια γενική συμπεριφορά στις λύσεις καθημερινών ανθρώπινων προβλημάτων, ή κάπως πιο ριζοσπαστικά, ότι όλες οι υποθέσεις βασίζονται σε παρελθοντικές αντίστοιχες περιπτώσεις που έχουμε βιώσει προσωπικά. Αυτή η άποψη σχετίζεται με τη θεωρεία πρωτοτύπου, που συναντάται κυρίως στην γνωστική επιστήμη.

1.3 Xml

Η XML είναι markup γλώσσα για έγγραφα που περιέχουν δομημένες πληροφορίες.Markup γλώσσα είναι ένας μηχανισμός που καθορίζει δομές σε ένα έγγραφο .Οι δομημένες πληροφορίες περιλαμβάνουν περιεχόμενο και κάποιες διευκρινίσεις για το ρόλο που παίζει το περιεχόμενο.Σχεδόν όλα τα έγγραφα έχουν την ίδια δομή.

Η XML είναι κάτι περισσότερο από markup language είναι metalanguage, δηλαδή μια γλώσσα που χρησιμοποιείται για να καθορίσει νέες markup γλώσσες.Η XML συμπληρώνει και δεν αντικαθιστά την HTML. Ενώ η HTML χρησιμοποιείται στη διατύπωση και την εμφάνιση των δεδομένων η XML αναπαριστά τη συναφή έννοια των δεδομένων.Στην HTML τα tags είναι προκαθορισμένα ενώ η XML παρέχει τη δυνατότητα να καθορίζουν οι χρήστες τα tags και τις δομημένες μεταξύ τους σχέσεις.

Τα XML έγγραφα δεν είναι πολύπλοκα αλλά απλά και πολύ αποτελεσματικά. Το διδακτικό υλικό της well-formed XML αναλύει τη δημιουργία των XML εγγράφων, η οποία είναι κατά κάποιο τρόπο ίδια με την HTML καθώς επιτρέπει τη μη δομημένη δημιουργία εγγράφου. Η valid XML είναι πιο σύνθετη. Απαιτεί την ύπαρξη ενός Document Type Definition πριν να γραφεί το έγγραφο αλλά παρέχει μια γενική δομή με βάση την οποία τη δημιουργούμε.

Η γλώσσα προγραμματισμού XML περιγράφει μια κατηγορία πληροφοριών (data objects) που καλούνται XML έγγραφα (documents) καθώς επίσης περιγράφει τμηματικά τη συμπεριφορά των προγραμμάτων που τα επεξεργάζονται.

Τα XML έγγραφα αποτελούνται από μονάδες αποθήκευσης που καλούνται *entities (οντότητες)*, οι οποίες περιέχουν πληροφορίες αναλυμένες ή μη. Οι αναλυμένες πληροφορίες αποτελούνται από *χαρακτήρες (characters)* οι οποίοι συνθέτουν *character data* και άλλοι οι οποίοι συνθέτουν *markup*. Η μορφή markup κωδικοποιεί την περιγραφή της τελικής αποθήκευσης του εγγράφου καθώς και τη λογική δομή.

Παραδειγμα Xml

<catalog>

Σχόλιο [DAD1]: Opening Tag

```
<product dept="WMN">
<number>557</number>
<name language="en">Fleece Pullover</name>
<colorChoices>navy black</colorChoices></product>
</catalog>
```

Σχόλιο [DAD2]: Attribute Node

Σχόλιο [DAD3]: Closing Tag

1.4 Xpath

Η XPath είναι μια γλώσσα για αναζήτηση πληροφοριών σε ένα έγγραφο XML Την XPath μπορούμε να την χρησιμοποιήσουμε για να έχουμε μια επιτυχή περιήγηση σε στοιχειά και χαρακτηριστικά ενός έγγραφου XML.Η XPath είναι ένα σημαντικό στοιχειό του W3C XSLT προτύπου.Δεν είναι τυχαίο πως και η XQuery και η XPointer βασίζονται σε εκφράσεις XPath.Έτσι η κατανόηση της XPath σου δίνει μεγάλα πλεονεκτήματα ως προς την χρησιμοποίηση της XML.

Παράδειγμα XPath

```
doc("catalog.xml")//product
```

1.5 Xquery

Η XQuery ή αλλιώς XML Query περιγράφει μια γλώσσα αναζήτησης βάσεων δεδομένων για δεδομένα XML.Ο καλύτερος τρόπος να εξηγήσεις τι είναι η XML είναι να πεις ότι είναι το αντίστοιχο της SQL σε έγγραφα XML.Η XQuery 1.0 δημιουργήθηκε από το XML Query working group του διεθνή οργανισμού W3C.Η ανάπτυξη της ήταν συνδεδεμένη με την ανάπτυξη της XSLT 2.0 από το XSL Working Group.Τα δυο group είναι υπεύθυνα για την δημιουργία της XPath 2.0 που είναι υποσύνολο της XQuery 1.0.Η XQuery 1.0 έγινε W3C πρότυπο τον Ιανουάριο του 2007.

Παράδειγμα XQuery

```
for $prod in doc("catalog.xml")/catalog/product
where $prod/@dept = "ACC"
order by $prod/name
return $prod/name
```

Σχόλιο [DAD4]: Παράδειγμα XQuery με FLWORs expression

1.6 Exist

Η Exist-db είναι ένα ανοιχτού κώδικα σύστημα διαχείρισης βάσεων δεδομένων που βασίζεται εξολοκλήρου στην τεχνολογία XML.Αποθηκεύει XML έγγραφα με βάση το XML data model.

Η Exist-db υποστηρίζει πολλά διαδικτυακά τεχνολογικά πρότυπα όπως:

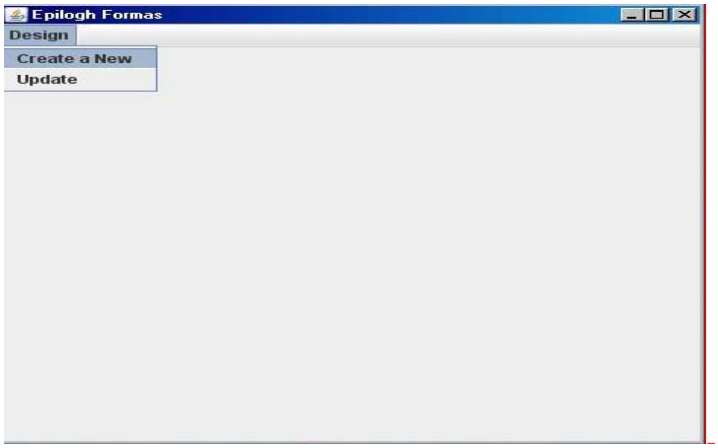
- XQuery 1.0 / XPath 2.0
- XSLT 1.0 (χρησιμοποιώντας Apache Xalan) or XSLT 2.0 (προαιρετική χρησιμοποίηση Saxon)
- HTTP interfaces: REST, WebDAV, SOAP, XMLRPC, Atom Publishing Protocol
- XML database specific: XMLDB, XQJ/JSR-225, XUpdate, XQuery update extensions

Η Exist-db παρέχει ένα ισχυρό περιβάλλον για την ανάπτυξη διαδικτυακών εφαρμογών που βασίζονται στην XQuery και στα σχετικά πρότυπα.

2 ΚΕΦΑΛΑΙΟ

Λειτουργικότητα

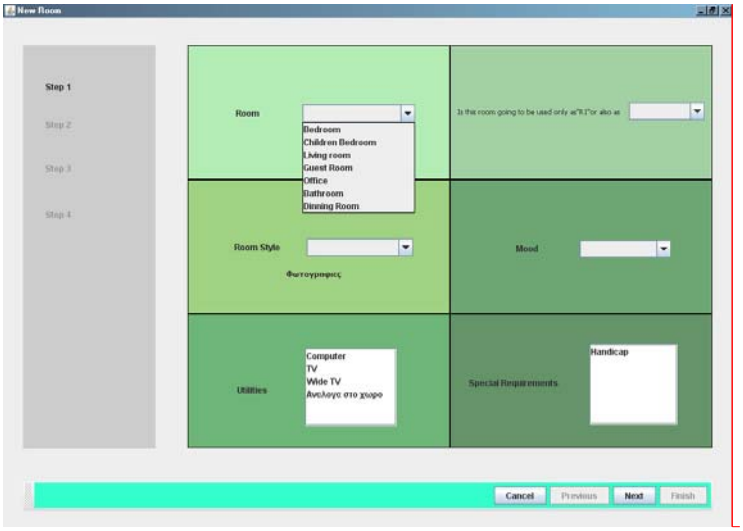
Ξεκινώντας την εφαρμογή μας ο χρηστής επιλέγει να δημιουργήσει ένα νέο δωμάτιο επιλέγοντας create new από το menu design όπως φαίνεται στην παρακάτω φωτογραφία.



Σχόλιο [DAD5]: Eik 1:Δημιουργια νέου χώρου.

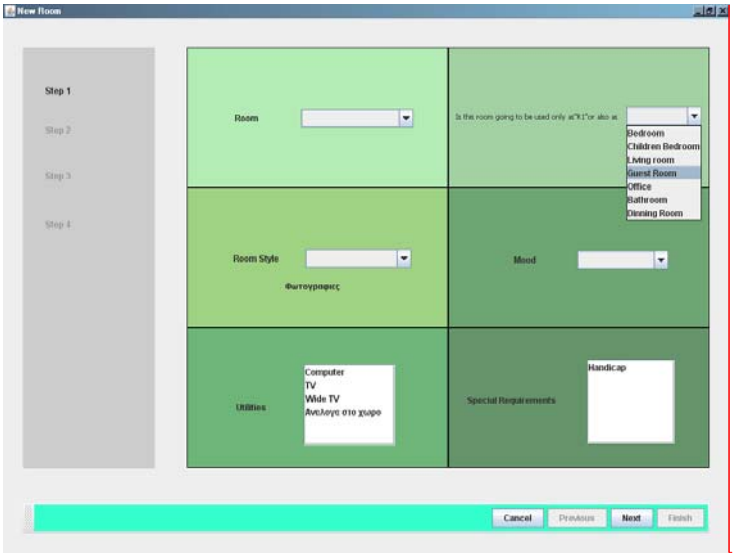
Στο επόμενο βήμα ο χρήστης θα αποτυπώσει τις επιθυμίες του και τις προτιμήσεις του για το σχεδιασμό ενός νέου χώρου.

Αρχικά ο χρηστής επιλεγεί την προτίμηση του στην καρτέλα Room(εικ2).Σ'αυτην την καρτέλα ο χρήστης επιλέγει την κύρια λειτουργία του χώρου.Δηλαδη αν θέλει να τον χρησιμοποιήσει ως Bedroom,Children Bedroom,Living room, Guest Room, Office, Bathroom, Dinning Room.Μια από τις παραπάνω επιλογές έχει το δικαίωμα να επιλέξει.



Σχόλιο [DAD6]: **Εικ 2.Επιλόγη κύριας λείτουργιας του χώρου.**

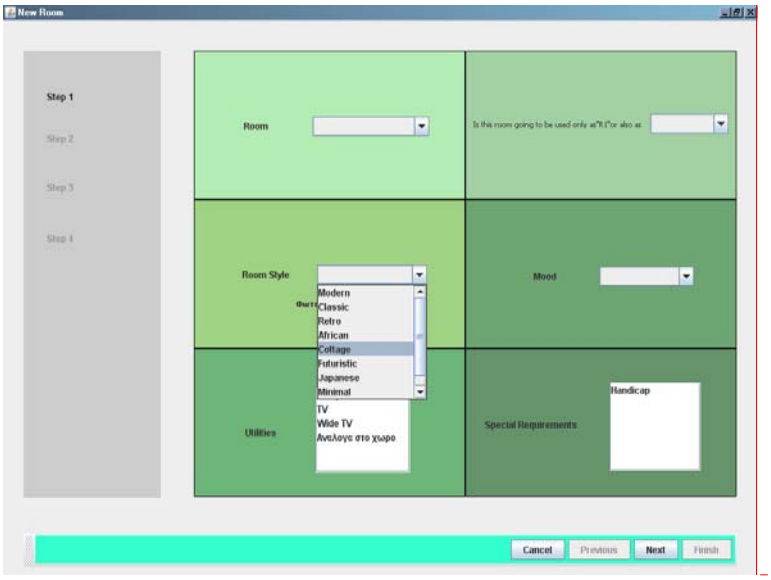
Στην συνέχεια επιλεγούμε την προτίμηση μας στην καρτέλα 'Is this room going to be used only as"R1"or also as '(εικ3).Σ'αυτην την καρτέλα ο χρήστης επιλέγει την δευτερεύοντα λειτουργία του χώρου.Δηλαδή αν θέλει να τον χρησιμοποιήσει επιπλέον ως Bedroom,ChildrenBedroom,Living room,Guest Room, Office, Bathroom, Dinning Room.Μια από τις παραπάνω επιλογές έχει το δικαίωμα να επιλέξει.



Σχόλιο [DAD7]: **Εικ 3.Επιλόγη δευτερεύον λειτουργεία του χώρου.**

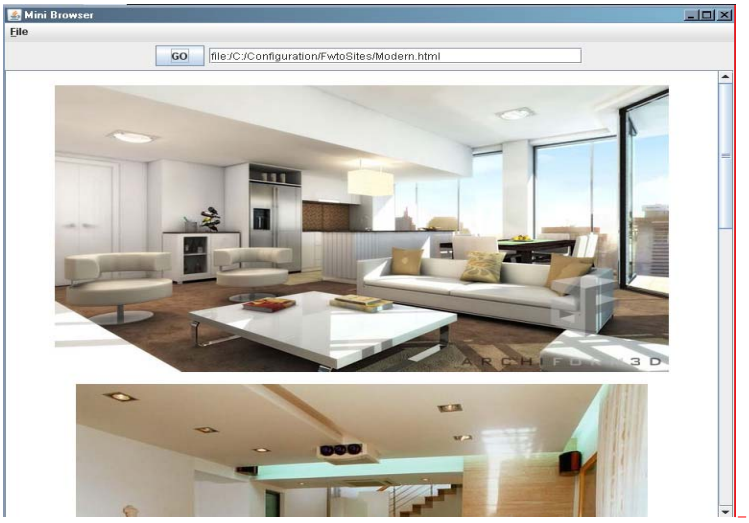
Επειτα στην καρτέλα RoomStyle ο χρηστής επιλέγει το Style που θα έχει ο συγκεκριμένος χώρος(εικ4).Οι επιλογές που έχουμε είναι οι έξης και ο χρήστης έχει

το δικαίωμα να διαλέξει μια από αυτές: Modern, Classic, Retro, African, Cottage, Futuristic, Japanese, Minimal,Morocco.



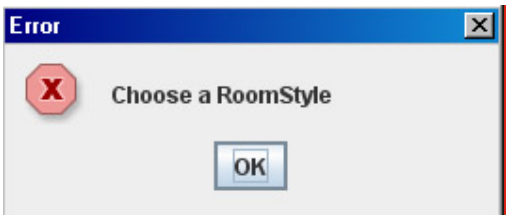
Σχόλιο [DAD8]: Εικ 4.Επιλογή Style

Όπως μπορούμε να δούμε κάτω απ την καρτέλα RoomStyle υπάρχει μια επιπλέον επιλογή Φωτογραφίες.Πατώντας αυτήν την καρτέλα ο χρηστής έχει την δυνατότητα να δει κάποια δείγματα φωτογραφιών για κάθε Style χωριστά.Οι φωτογραφίες απεικονίζονται σ'ένα Browser(εικ5).



Σχόλιο [DAD9]: Εικ 5:Απεικόνιση δείγματα φωταγραφιών για κάθε Style χωριστά μεσω Browser.

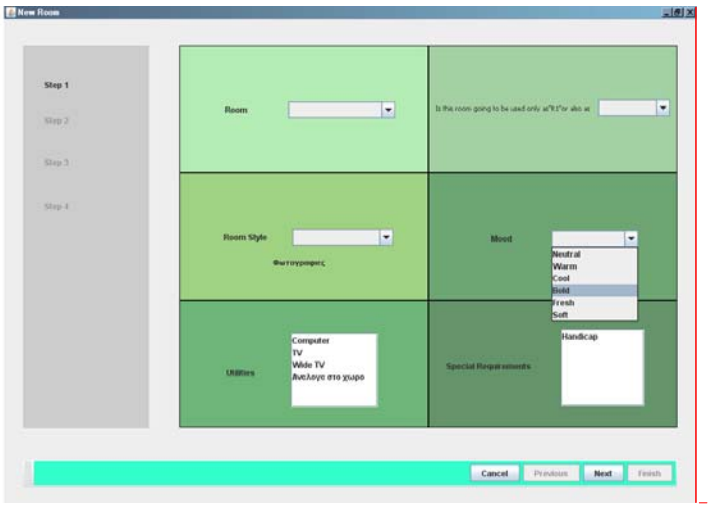
Αξίζει να σημειωθεί ότι αν ο χρήστης επιλέξει την καρτέλα φωτογραφίες δίχως να διαλέξει μια επιλογή από την καρτέλα RoomStyle το σύστημα θα τον προειδοποιήσει ότι υπάρχει λάθος με το παρακάτω μήνυμα(εικ6).



Σχόλιο [DAD10]: Εικ 6:Μήνυμα λαθούς

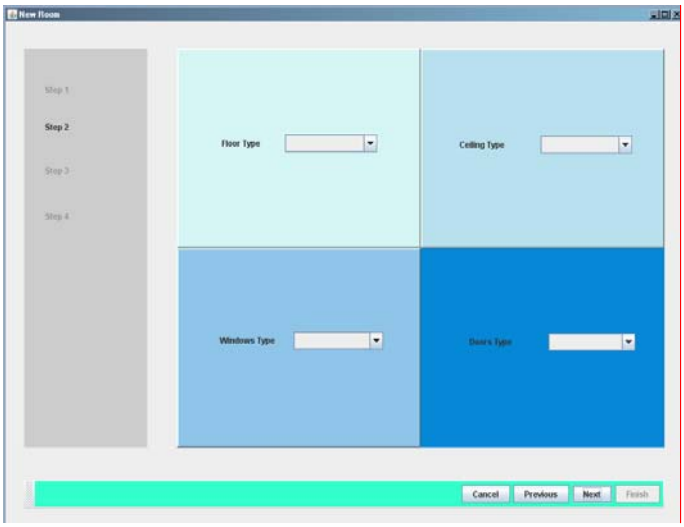
Αυτό σημαίνει ότι για να μπορέσει να δει κάποια δείγματα φωτογραφιών για κάποιο Style θα πρέπει πρώτα να επιλέξει ένα συγκεκριμένο Style και έπειτα να πατήσει στην καρτέλα φωτογραφίες.

Η επόμενη επιλογή είναι ο χρήστης να αποφασίσει για την διάθεση που θα θέλει να έχει ο συγκεκριμένος χώρος.Έτσι στην καρτέλα Mood (εικ7) θα πρέπει να επιλέξει ανάμεσα σε Neutral, Warm, Cool, Bold, Fresh,Soft.Αυτη η επιλογή συνδέεται άμεσα με την επιλογή ColourToAvoid που θα συναντήσει παρακάτω ο χρήστης.Στο τέλος αυτής της καρτέλας ο χρηστής έχει την δυνατότητα να επιλέξει κάποια Utilities για τον συγκεκριμένο χώρο και επίσης να επισημάνει στο σύστημα αν ο άνθρωπος που θα μείνει στον συγκεκριμένο χώρο έχει κάποια κινητικά προβλήματα.



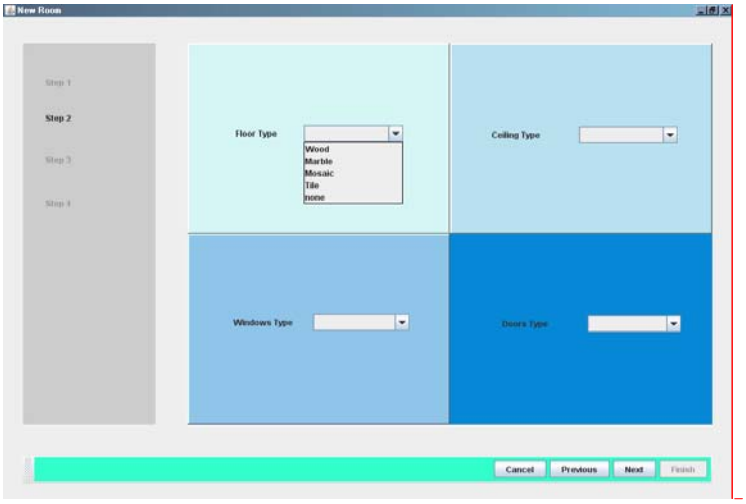
Σχόλιο [DAD11]: Εικ 7:Επιλογή διάθεσης που θέλουμε να έχει ο σγκεκρίμενος χώρος.

Πατώντας το κουμπί next ο χρηστής πηγαίνει στην επόμενη καρτέλα και βλέπει τις εξής επιλογές(εικ8).Σ'αυτό το βήμα ο χρήστης θα είναι σε θέση να επιλέξει τα υλικά απ τα οποία θα είναι φτιαγμένος κάποια συγκεκριμένα μέρη του χώρου.



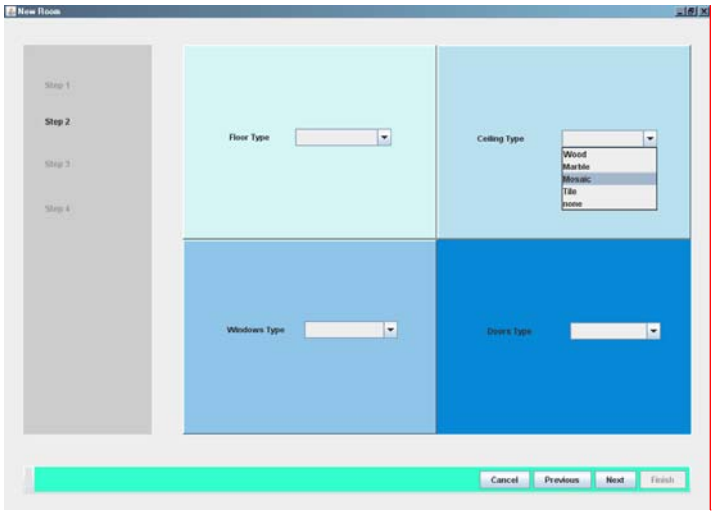
Σχόλιο [DAD12]: Εικ 8:Επιλογή υλικών απ'τα οποία θα είναι φτιαγμένος ο συγκεκρίμενος χώρος.

Αρχικά ο χρήστης θα πρέπει να επιλέξει το είδος του υλικό απ τό όποιο θα είναι φτιαγμένο το πάτωμα.(Floor Type).Θα πρέπει να επιλέξει ανάμεσα σε: Wood, Marble,Mosaic,Tile,none.(εικ9)



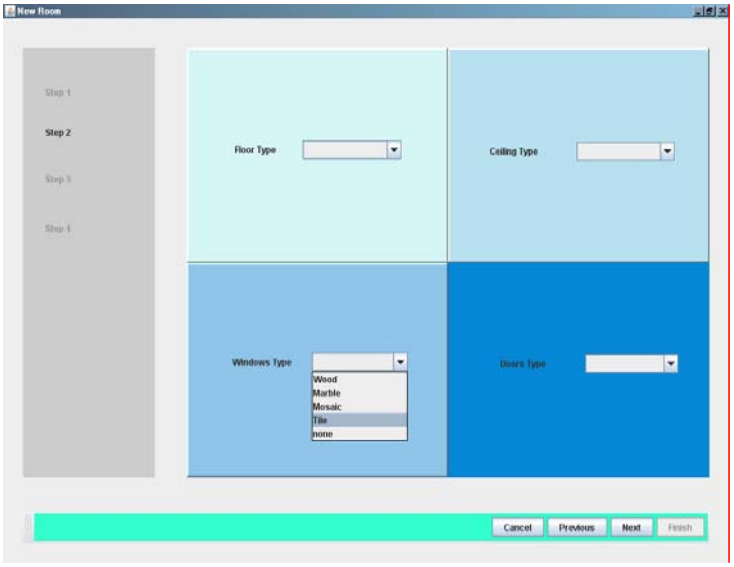
Σχόλιο [DAD13]: Εικ 9: Επιλογή του υλικού απ'το οποιο θα είναι φτιαγμένο το πάτωμα

Έπειτα ο χρήστης θα πρέπει να επιλέξει το είδος του υλικού απ τό όποιο θα είναι φτιαγμένο το νταβάνι.(Ceiling Type).Θα πρέπει να επιλέξει ανάμεσα σε: Wood, Marble,Mosaic,Tile,none.(εικ10)



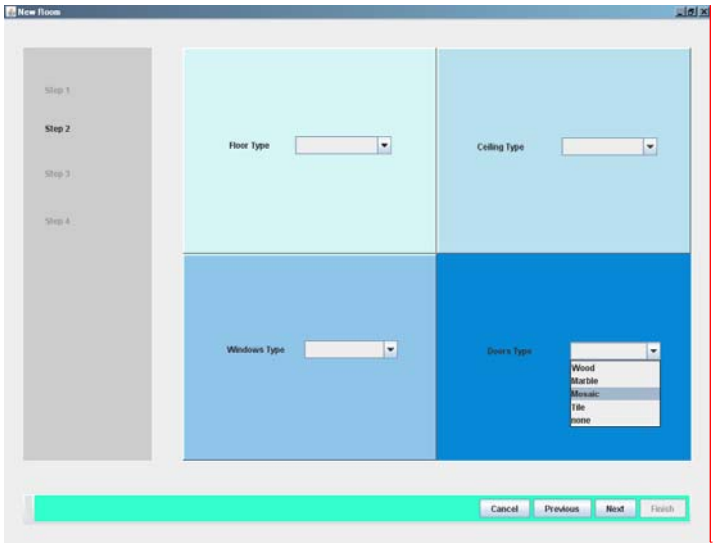
Σχόλιο [DAD14]: Εικ10: Επιλογή του υλικού απ'το οποιο θα είναι φτιαγμένο το νταβάνι

Στην συνεχεία ο χρήστης θα πρέπει να επιλέξει το είδος του υλικού απ το όποιο θα είναι φτιαγμένο το παράθυρο ή τα παράθυρα.(Windows Type).Θα πρέπει να επιλέξει ανάμεσα σε: Wood, Marble,Mosaic,Tile,none.(εικ11)



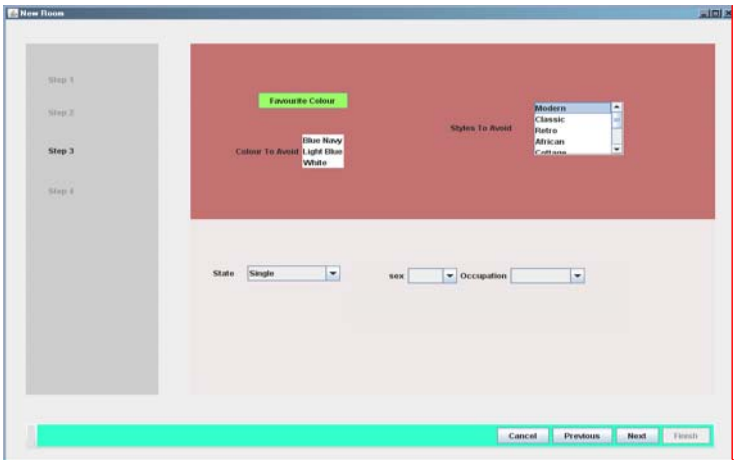
Σχόλιο [DAD15]: Εικ11: Επιλογή του υλικού απ'το οποιο θα είναι φτιαγμενο το παράθυρο ή τα παράθυρα.

Τέλος ο χρήστης θα πρέπει να επιλέξει το είδος του υλικού απ τό όποιο θα είναι φτιαγμένο η πόρτα ή οι πόρτες.(Doors Type).Θα πρέπει να επιλέξει ανάμεσα σε: Wood, Marble,Mosaic,Tile,none.(εικ12)



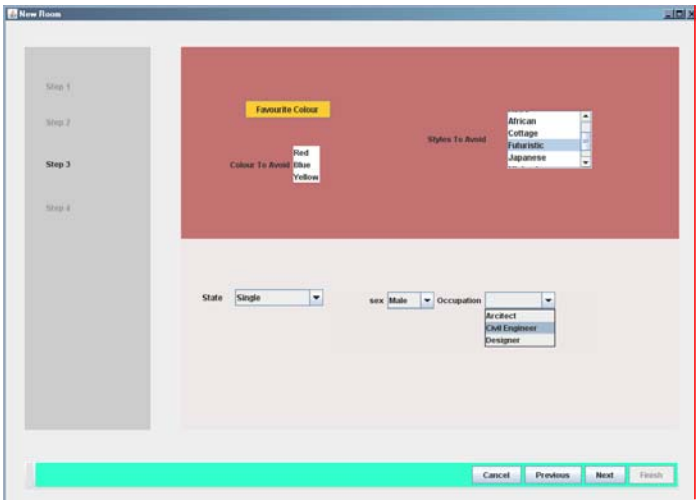
Σχόλιο [DAD16]: Εικ12: Επιλογή του υλικού απ'το οποιο θα είναι φτιαγμενη η πόρτα.

Πατώντας next πηγαίνουμε στο επόμενο βήμα στο όποιο ο χρήστης θα μπορεί να αποφασίσει για το τι χρώμα επιθυμεί να έχει ο χώρος,το τι χρώμα και τι Style θα ήθελε να αποφύγει και τέλος θα πρέπει να δώσει πληροφορίες για την οικογενειακή του κατάσταση.(εικ13).Αξίζει να σημειωθεί ότι η επιλογή Color To Avoid είναι άμεσα συνδεδεμένη με την επιλογή Mood που είδαμε στην πρώτη καρτέλα. Ανάλογα τι θα επιλέξει στο Mood,διαφορετική τριάδα χρωμάτων θα εμφανιστεί για επιλογή στο Colour To Avoid.

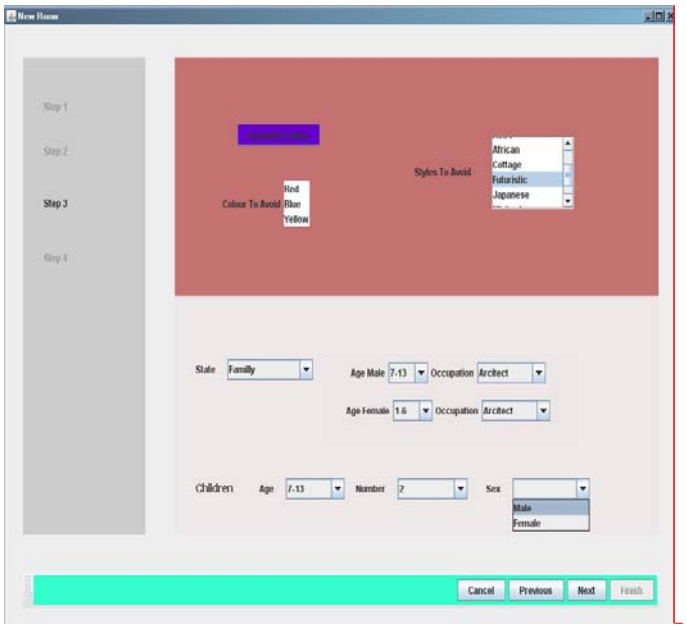


Σχόλιο [DAD17]: Eik13:Επιλογή Colour,Colour To Avoid,Style To Avoid και Οικογενειακη του κατάσταση.

Όσο αναφορά την οικογενειακή του κατάσταση ο χρήστης θα πρέπει να επιλέξει αν είναι παντρεμένος με παιδιά ή ελεύθερος. Ανάλογα τι θα επιλέξει στην επιλογή State διαφορετικά πεδία θα εμφανιστούν.(εικ14,εικ15)



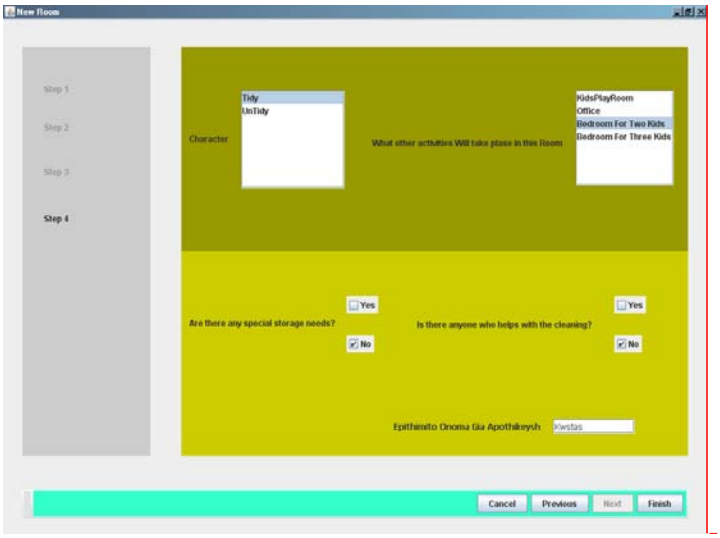
Σχόλιο [DAD18]: Εικ14:Επιλογή Single στο πεδιο State.



Σχόλιο [DAD19]: Εικ15:Επιλογή Familly στο πεδιο State.

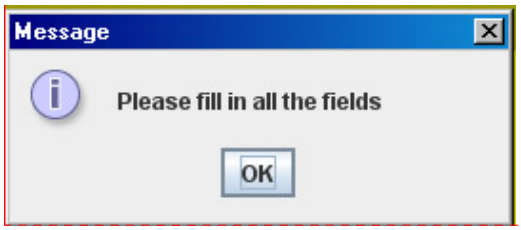
Πατώντας το κουμπί next πηγαίνουμε στην επομένη καρτέλα στην οποία ο χρήστης θα πρέπει να δώσει κάποιες πληροφορίες για τον χαρακτήρα του και αν ο συγκεκριμένος χώρος θα τον χρησιμοποιήσει και για άλλες δραστηριότητες. Ακόμα

θα πρέπει να δηλώσει αν θα υπάρχει κάποιος που θα βοηθάει με το καθάρισμα και τέλος θα πρέπει να δηλώσει το όνομα που επιθυμεί να έχει το XML έγγραφο στο όποιο θα αποθηκευτούν οι πληροφορίες που μας έδωσε.(εικ16).



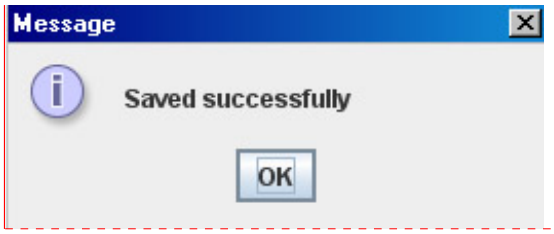
Σχόλιο [DAD20]: Εικ16:Επιλογή Character, Other activities Will take plase in this Room, Any special storage, Is there anyone who helps with the cleaning και επιθύμιτο όνομα για αποθήκευση.

Μόλις ο χρηστής πατήσει το κουμπί finish όλες οι πληροφορίες που μας έδωσε θα αποθηκευτούν σε ένα XML έγγραφο με το όνομα που δήλωσε αλλά επίσης και την ημερομηνία και την ώρα που έκανε την αποθήκευση. Αυτό το xml παίρνει την μορφή από ένα άλλο xml που υπάρχει στην Exist.Αυτό το xml έχει όνομα Exist_Template και περιέχει μόνο τα κύρια χαρακτηριστηκά που έχει ο κάθε χώρος.Πριν το Xml αποθηκευτεί με τις επιθυμίες του χρηστή στην Exist δημιουργείται αυτόματα ένα copy από αυτό στο δίσκο με όνομα στο XmlCopy.xml στο φάκελο C:\Configuration.Επειτα αυτό το xml αποθηκεύεται στην Exist με αποτυπωμένες τις επιθυμίες του χρηστή. Αξίζει να σημειωθεί ότι πρέπει ο χρήστης να έχει συμπληρώσει όλα τα πεδία που του έχουν ζητηθεί..Εάν θα έχει αφήσει κάποιο κενό το σύστημα μας θα τον πληροφορήσει γι αυτό με το παρακάτω μήνυμα(εικ17):



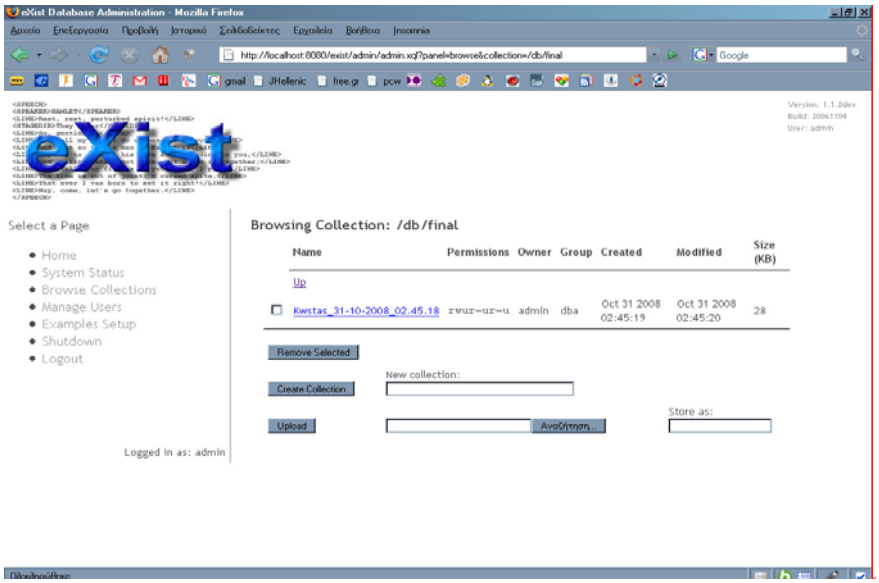
Σχόλιο [DAD21]: Εικ17:Μήνυμα λάθους εάν ο χρηστής δεν έχει επιλεξει όλα τα πεδια.

Εαν όλα έχουν πάει καλά τότε το σύστημα θα τον πληροφορήσει με το παρακάτω μήνυμα :(εικ18)



Σχόλιο [DAD22]: **Εικ18:Μήνυμα για επιτυχημένη αποθήκευση.**

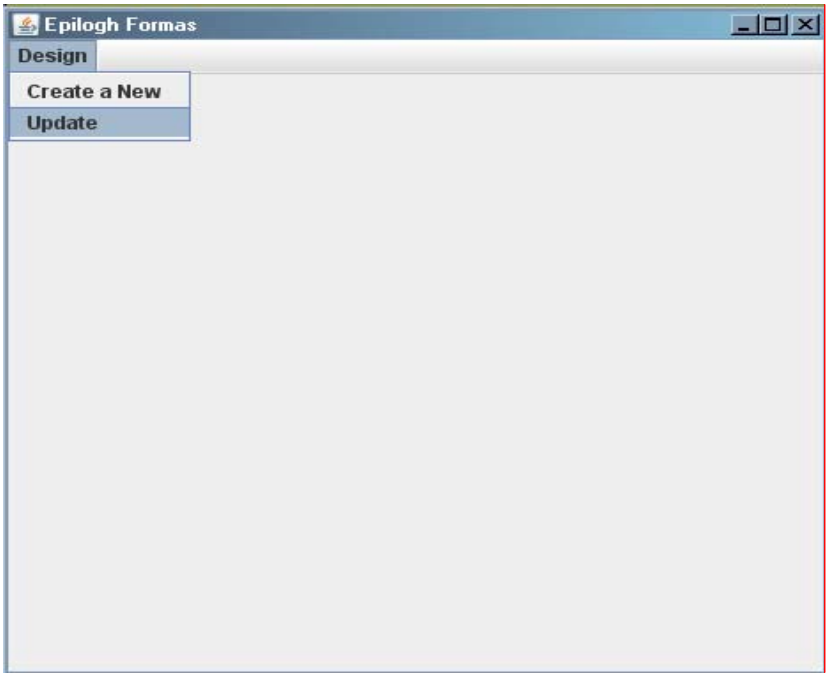
Για παράδειγμα εάν έχουμε δηλώσει το όνομα Κώστας ως επιθυμητό όνομα για αποθήκευση βάση δεδομένων μας (Eexist) θα έχει την παρακάτω μορφή:



Σχόλιο [DAD23]: **Εικ 19:Μορφή της Exist μετά από πετυχημένη αποθήκευση.**

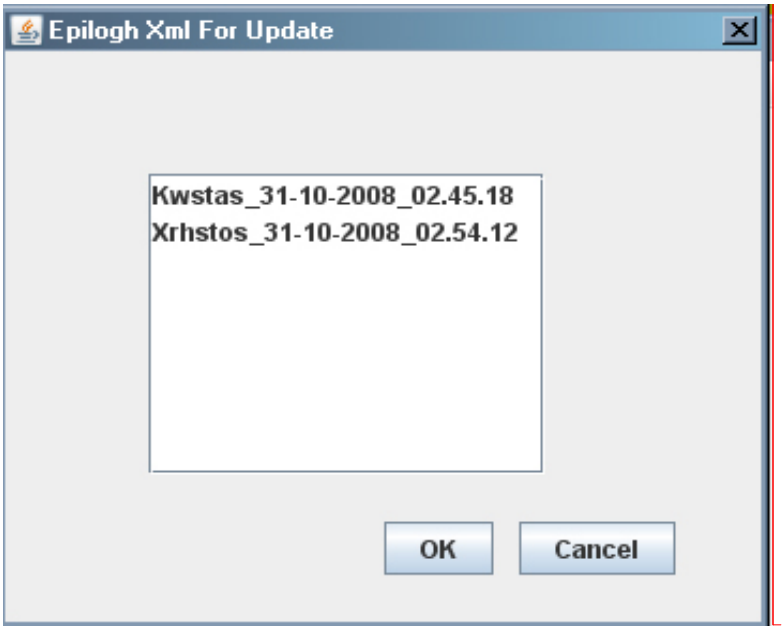
Τώρα αν θελήσει ο χρήστης οποιαδήποτε στιγμή θέλει να κάνει κάποια αλλαγή στις επιλογές που διάλεξε το μόνο που θα πρέπει να κάνει είναι να διαλέξει την επιλογή Update απ την αρχική μας φόρμα:

Μόλις επιλέξει Update θα εμφανιστεί η παρακάτω φόρμα:



Σχόλιο [DAD24]: Εικ 20:Επιλογή Update απ την αρχική μας φόρμα.

Σ'αυτό το βήμα ο χρήστης θα παρατηρήσει κάποια ονόματα συνδιασμένα με ημερομηνίες και ώρες.



Σχόλιο [DAD25]: Εικ21:Λίστα με τα ονόματα των XML που είναι αποθηκευμένα στην Βάση Δεδομένων μας την δεδομένη χρονική στιγμή(Exist).

Αυτά τα ονόματα δεν είναι τπτ άλλο από τα ονόματα των XML εγγράφων που είναι αποθηκευμένα στην βάση δεδομένων μας.(Exist-db).Ανάμεσα σ'αυτά τα ονόματα ο

χρηστής θα δει και το όνομα που ο ίδιος είχε δηλώσει στην διαδικασία της δημιουργίας καινούργιου χώρου (προηγούμενη διαδικασία).Μόλις επιλέξει το όνομα του ο χρήστης θα δει τις προήγουμενες καρτέλες με την μονή διάφορα ότι τα πεδία θα είναι γεμάτα με τις επιλογές τις οποίες ο ίδιος διάλεξε στην διαδικασία της δημιουργίας καινούργιου χώρου. Τώρα ο χρήστης μπορεί να αλλάξει αν θέλει κάτι και να διαλέξει κάποια άλλη επιλογή με τον ίδιο τρόπο που χρησιμοποιήσε στην διαδικασία δημιουργίας καινούργιου χώρου. Ακόμα μια διαφορά σ'αυτές τις καρτέλες είναι ότι ο χρήστης δεν μπορεί να δηλώσει νέο ονομα.Το Update θα γίνει στο ήδη υπάρχον από πριν Xml έγγραφο στην Exist.Μόλις πατηθεί το Update και εφόσον όλα θα έχουν πάει καλά το σύστημα θα πληροφορήσει τον χρήστη με το παρακάτω μήνυμα.

Σχόλιο [DAD26]: Εικ22:Μήνυμα συστήματος που μας πληροφορεί ότι οι επιλογες μας ανανεώθηκαν.

Εάν ο χρήστης θελήσει να δει τις αλλαγές του μπορεί να ξανακάνει την διαδικασία του Update όπως πριν και θα παρατηρήσει ότι στις καρτέλες που δηλώνει τις επιθυμίες του οι επιλογές που είχε κάνει στην διαδικασία της δημιουργίας καινούργιου χώρου θα έχουν πλέον αλλάξει.Αξιζει να σημειωθεί ότι την διαδικασία του Update μπορεί να την κάνει όσες φορές θέλει.Αν θέλει απλά να δει τις αλλαγές χωρίς να κάνει επιπλέον άλλες αλλαγές μπορεί να βγει από το σύστημα πατώντας το κουμπί cancel.

3 ΚΕΦΑΛΑΙΟ

Τεχνολογία

Ξεκινάμε από το NetBeans το πρώτο JFrame με όνομα Αρχική.java.Κοιτώντας το κώδικα του βλέπουμε ότι εκτός απ την main έχουμε και δυο ActionListeners(Ενα για κάθε JMenuItem του JFrame)και την μέθοδο initComponents().

3.1 Ανάλυση της Κλάσης Αρχική.java

Ο πρώτος ActionListener αναφέρεται στο JMenuItem με όνομα Create a New.

3.1.1 Ο κώδικας του ActionListener του jMenuItem1:

```
private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {

    MyWizard newa=new MyWizard(); (1)

    newa.setVisible(true); (2)

    this.dispose(); (3)

}
```

(1) Δημιουργούμε ένα αντικείμενο τύπου MyWizard

(2) Το κάνουμε Visible

(3) Κλείνουμε το υπάρχον JFrame (Arxikh.java)

Ο δεύτερος ActionListener αναφέρεται στο JMenuItem με όνομα Update.

3.1.2 Ο κώδικας του ActionListener jMenuItem2:

```
private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {

    LoadMyXml xmlupdate=new LoadMyXml(this); (1)

    xmlupdate.setVisible(true); (2)}
```

(1) Δημιουργούμε ένα αντικείμενο τύπου LoadMyXml και του περνάμε σαν όρισμα το JFrame στο όποιο βρισκόμαστε.

(2) Το κάνουμε Visible

Αξίζει να σημειωθεί ότι το αντικείμενο LoadMyXml είναι JDialog (γι'αυτό παίρνει και σαν όρισμα JFrame)και το αντικείμενο MyWizard είναι JFrame.

3.2 Ανάλυση της κλάσης LoadMyXml.java

```
public LoadMyXml(java.awt.Frame parent) {

    super(parent,true); (1)

    initComponents(); (2)

    filljList(); (3)

  }
```

(1) Κάνουμε modal το JDialog

(2) Καλουμε την initComponents() να εκτελεστεί η οποία είναι υπεύθυνη για το σχεδιασμό της LoadMyXml.

(3) Καλουμε την filljList();

3.2.1 Επεξήγηση της μεθόδου filljList()

```
private void filljList() {

   try{//Σύνδεση με την Exist

String driver = "org.exist.xmldb.DatabaseImpl";

            Class cl = Class.forName(driver);

            Database database = (Database) cl.newInstance();

            DatabaseManager.registerDatabase(database);
```

```
//Παίρνουμε την Collection που θέλουμε και την αποθηκεύουμε σε μια μεταβλητή τύπου Collection με όνομα col.

Collection col=DatabaseManager.getCollection("xmldb:exist://localhost:8080/exist/xmlrpc/db/final");

//Παίρνουμε την λίστα με τις XMLResources της Collection col και τις αποθηκεύουμε σε ένα αντικείμενο τύπου JList με όνομα jList1.

      String [] collections=col.listResources();

 int i=0;

while (i<collections.length){

     addList.add(i,collections[i]);

 jList1.setListData(addList);

 i++; }}  catch(Exception e1){ e1.printStackTrace();}}

//Αυτό που θέλουμε να πετύχουμε μ'αυτην την μέθοδο είναι πάρουμε τα ονόματα των XMLResources από μια συγκεκριμένη Collection που είναι αποθηκευμένη στην Exist και να τα εμφανίσουμε μέσα σε μια JList.
```

3.2.2 Επεξήση του ActionListener του κουμπιού jButton1

```
  private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

//Αποθηκεύουμε το αποτέλεσμα της επιλογής του jList1 σε μια μεταβλητή τύπου String με όνομα xml.Επισης ελέγχουμε αν ο χρήστης δεν έχει επιλέξει καμία επιλογή από την JList.Αν ισχύει κάτι τέτοιο τότε το σύστημα μας πληροφορεί με το ανάλογο μήνυμα.

       String xml=(String)jList1.getSelectedValue();

       if(jList1.isSelectionEmpty()==true){
```

```
JOptionPane.showMessageDialog(this, "Epilejte ena arxeio XML apo thn lista");  }else{

//Δημιουργούμε ένα αντικείμενο τύπου Update με όρισμα την επιλογή που έκανε ο χρήστης από την JList.Επίσης κλείνουμε την LoadMyXml.java αλλά και την Arxikh.java.

Update upxml=new Update(xml);

      upxml.setVisible(true);

      this.dispose();

    Window b= (Window) this.getParent();

    b.dispose(); } }
```

Αυτό που θέλουμε να κάνουμε μ'αυτην την μέθοδο είναι να ελέγξουμε αν ο χρήστης έχει επιλέξει μια επιλογή από την λίστα και αν ναι να πάρουμε αυτήν την επιλογή και να την περάσουμε σαν όρισμα στην κλάση Update δημιουργώντας ταυτόχρονος ένα αντικείμενο τύπου Update.Έτσι με αυτό τον τρόπο θα επιτύχουμε να βλέπουμε στην φόρμα Update για κάθε χρηστή χωριστά με τις επιλογές που έχει κάνει.

3.3 Ανάλυση της Κλάσης MyWizard

```
public MyWizard() {

//Καλούμε την initComponents() που είναι υπεύθυνη για το σχεδιασμό της κλάσης MyWizard().Επίσης δίνουμε τιμές σε δυο μεταβλητές (next,check) και κάνουμε enable κάποια JButtons και JLabels που θέλουμε να είναι ενεργά όταν ξεκάνει η εφαρμογή μας και αντίστοιχα κάνουμε disable κάποια JButtons και JLabels που θέλουμε να είναι μη ενεργά όταν ξεκάνει η εφαρμογή μας.

initComponents();

next=0;check=3;check2=3;
```

```
jButton2.setEnabled(false);

    jButton4.setEnabled(false);

        jLabel2.setEnabled(false);

         jLabel3.setEnabled(false);

          jLabel4.setEnabled(false);
```

//Όλες οι παρακάτω δηλώσεις αφορούν το σημείο που ο χρήστης επιλεγεί την οικογενειακή του κατάσταση.Ανάλογα τι θα διαλέξει θα εμφανίζονται στην φόρμα και οι ανάλογες επιλογες.Ολες αυτές οι επιλογές αρχικά δεν θα είναι εμφανείς.

```
 jl1 = new javax.swing.JLabel("sex");

      jPanel22.add(jl1);

  j1=new javax.swing.JComboBox();

      jPanel22.add(j1);

     jl4 = new javax.swing.JLabel("Occupation");

      jPanel22.add(jl4);

      j2=new javax.swing.JComboBox();

       jPanel22.add(j2);

    jl2 = new javax.swing.JLabel("Age Male");

      jPanel22.add(jl2);

  j3=new javax.swing.JComboBox();

       jPanel22.add(j3);

   jl5 = new javax.swing.JLabel("Occupation");
```

```
        jPanel22.add(jl5);
 j4=new javax.swing.JComboBox();
        jPanel22.add(j4);
      jl3 = new javax.swing.JLabel("Age Female");
jPanel23.add(jl3);
   j5=new javax.swing.JComboBox();
        jPanel23.add(j5);
   jl6 = new javax.swing.JLabel("Occupation");
       jPanel23.add(jl6);
       j6=new javax.swing.JComboBox();
        jPanel23.add(j6);
          jlmood=new JLabel("Colour To Avoid");
      jPanel24.add(jlmood);
        jmood1=new JList();
        jmood1.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
      jPanel24.add(jmood1);
         j1.setVisible(false);
       j2.setVisible(false);
       j3.setVisible(false);
         j4.setVisible(false);
         j5.setVisible(false);
```

```
 j6.setVisible(false);

          jl1.setVisible(false);

           jl2.setVisible(false);

              jl3.setVisible(false);

        jl4.setVisible(false);

        jl5.setVisible(false);

        jl6.setVisible(false);

        jmood1.setVisible(false);

     jlmood.setVisible(false);

      //Καλούμε την doFill() η οποία είναι υπεύθυνη για το γέμισμα όλων των
jComboBox και των JList της εφαρμογής μας.Το γέμισμα γίνεται δυναμικά από
ιεραρχικό Xml που βρίσκεται στον δίσκο μας με όνομα ComboBox.xml και περιέχει όλα
τα δεδομένα που χρειάζονται.

doFill();}
```

3.3.1 Επεξήγηση της μεθόδου doFill()

```
private void doFill() {

      try{try{ try{try{try{

//Διαβάζουμε το Xml από το δίσκο που έχει τα δεδομένα μας χρησιμοποιώντας έναν
FileReader.Έπειτα αποθηκεύουμε το αποτέλεσμα από αυτά που διαβάσαμε σε μια
μεταβλητή τύπου String.

     File f = new File("C:\\Configuration\\Combobox.xml");

   FileReader rd = new FileReader(f);
```

```
        char[] buf = new char[(int)f.length()];

        rd.read(buf);

        patternStr = new String(buf);

//Περνάμε τα δεδομένα σε μια μεταβλητή τύπου InputStream και δημιουργούμε ένα αντικείμενο τύπου DocumentBuilderFactory.Μέσω αυτού δημιουργούμε ένα αντικείμενο τύπου docBuilder και μέσω αυτού κάνουμε parse τα δεδομένα μας σε ένα αντικείμενο τύπου Document και με όνομα myXML.

InputStream is = new ByteArrayInputStream(patternStr.getBytes("utf-8"));

            DocumentBuilderFactory
docBuilderFactory=DocumentBuilderFactory.newInstance();

                DocumentBuilder docBuilder =docBuilderFactory.newDocumentBuilder();

                Document myXML =docBuilder.parse(is);

                myXML.normalize();

//Μεσω του myXML παίρνουμε το DocumentElement του XML μας στην περίπτωση μας το <Combo_Box> και το αποθηκεύουμε στην μεταβλητή myXMLElement.Έπειτα μέσω αυτού παίρνουμε μια λίστα με τα Elements που έχουν tagName Drop_Down και τα αποθηκεύουμε σε μια μια μεταβλητή τύπου Nodelist.

                 Element myXMLElement = myXML.getDocumentElement();

 NodeList commonList = myXMLElement.getElementsByTagName("Drop_Down");

            int i=0;

//Με βάση το μέγεθος της NodeList φτάνουμε στο attribute του κάθε tag με όνομα Drop_Down και παίρνουμε την τιμή του.

        while (i<commonList.getLength())

                {
```

```
 Vector <Object>w=new Vector<Object>();

  Node commonNode = commonList.item(i);

   Element commonElement = (Element)commonNode;

    String b=(String)commonElement.getAttribute("name");
```

//Ανάλογα το όνομα που θα έχει κάθε φορά η μεταβλητή b μπαίνει στο κατάλληλο loop και παίρνουμε μια λίστα με τα Elements που έχουν tagName Item και τα αποθηκεύουμε σε μια μια μεταβλητή τύπου Nodelist.

```
                if (b.equalsIgnoreCase("Label1")){

          NodeList  nodeList1=commonElement.getElementsByTagName("Item");

          int k=0;

      while (k<nodeList1.getLength()){

Node node1 = nodeList1.item(k);

                Element element1 = (Element)node1;

            Node myNode = node1.getNextSibling();

           myNode=element1.getFirstChild();

              String a =myNode.getNodeValue();

            jComboBox1.insertItemAt(a,k);

             k++;} }

                    i++; } }

               catch (FileNotFoundException e){

                  } }
```

```
 catch (java.io.IOException e){

            }}catch (ParserConfigurationException e){ }

} catch (org.xml.sax.SAXException e){}      }

catch(Exception e){e.printStackTrace();}   }

//Αυτος ο κώδικας επαναλαμβάνεται για όλα τα jComboBox και για όλα JList της εφαρμογής μας.
```

3.3.2 Επεξήγηση του ActionListener του κουμπιού jButton3 (κουμπί Next)

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

//Παίρνουμε το layout από το βασικό panel μας.Καθε φορά που πατάμε το κουμπί next της εφαρμογής μας η μεταβλητή next αυξάνεται κατά 1 και ανάλογα τι τιμή έχει πηγαίνει στην επομένη καρτέλα και γίνονται ενεργά και ανενεργά αντίστοιχα κάποια jButtons και jLabels.

CardLayout card=(CardLayout)jPanel1.getLayout();

      next++;

      if(next==1){

      card.next(jPanel1);

      jButton2.setEnabled(true);

      jLabel1.setEnabled(false);

      jLabel2.setEnabled(true);

 }else if(next==2){
```

```
            card.next(jPanel1);

        jLabel3.setEnabled(true);

        jLabel2.setEnabled(false); }

       else if(next==3){

            card.next(jPanel1);

          jButton4.setEnabled(true);

          jButton3.setEnabled(false);

           jLabel3.setEnabled(false);

           jLabel4.setEnabled(true); }}
```

//Ο κώδικας αυτός αναφέρεται στο κουμπί next της εφαρμογής μας και το αποτέλεσμα που μας δίνει κάθε φορά που θα πατηθεί είναι να αλλάζουν μια μια οι καρτέλες μας (jPanels) που βρίσκονται πάνω στο βασικό panel μας (jPanel1).

3.3.3 Επεξήγηση του ActionListener του κουμπιού jButton2 (κουμπί Previous)

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

//Παίρνουμε το layout από το βασικό panel μας.Καθε φορά που πατάμε το κουμπί previous της εφαρμογής μας η μεταβλητή next μειώνεται κατά 1 και ανάλογα τι τιμή έχει πηγαίνει στην προηγούμενη καρτέλα από εκεί που είναι και γίνονται ενεργά και ανενεργά αντίστοιχα κάποια jButtons και jLabels.

CardLayout card=(CardLayout)jPanel1.getLayout();

       next--;

    if(next==0){

       card.first(jPanel1);
```

```
        jButton2.setEnabled(false);

        jLabel1.setEnabled(true);

        jLabel2.setEnabled(false);}

      else if(next==1){

card.previous(jPanel1);

         jLabel3.setEnabled(false);

         jLabel2.setEnabled(true);}

         else if(next==2){

              card.previous(jPanel1);

            jButton4.setEnabled(false);

            jButton3.setEnabled(true);

             jLabel3.setEnabled(true);

             jLabel4.setEnabled(false);} }
```

//Ο κώδικας αυτός αναφέρεται στο κουμπί previous της εφαρμογής μας και το αποτέλεσμα που μας δίνει κάθε φορά που θα πατηθεί είναι να αλλάζουν προς τα πίσω μια μια οι καρτέλες μας (jpanels) που βρίσκονται πάνω στο βασικό panel μας (jPanel1).

3.3.4 Επεξήγηση της μεθόδου MouseClicked της jLabel9

```
private void jLabel9MouseClicked(java.awt.event.MouseEvent evt) {

//Ελέγχουμε αν έχουμε διαλέξει κάποια επιλογή από το jComboBox3.Αν όχι το σύστημα μας πληροφορεί με το ανάλογο μήνυμα.

        if (jComboBox3.getSelectedItem()==null){

            JOptionPane.showMessageDialog(this, "Epeleje RoomStyle",
```

```
"Error", JOptionPane.ERROR_MESSAGE);

//Αν έχουμε διαλέξει κάποια επιλογή τότε αυτήν την επιλογή την περνάμε σαν όρισμα τύπου String στο αντικείμενο τύπου MiniBrowser δημιουργώντας συγχρόνως ένα αντικείμενο τύπου MiniBrowser με όνομα app.

}else{

      String rs=(String)jComboBox3.getSelectedItem();

 MiniBrowser app = new MiniBrowser(rs);

//Τελος το εμφανίζουμε στο κέντρο της οθόνης.

      int frameWidth = 768;

            int frameHeight = 650;

            Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();

            app.setBounds((int) (screenSize.getWidth() - frameWidth)/2,
(int)(screenSize.getHeight()-frameHeight)/2, frameWidth, frameHeight);

    app.setVisible (true);}}
```

3.3.5 Επεξήγηση του ActionListener του κουμπιού Button4 (κουμπί Finish)

```
//Σ'αυτό το σημείο το σύστημα μας ελέγχει αν έχουν επιλεχτεί όλα τα jComboBox και jList που βρίσκονται στην εφαρμόγη μας.Άν δεν έχει γίνει κάτι τέτοιο το σύστημα μας ειδοποιεί με το ανάλογο μήνυμα.

if(jComboBox1.getSelectedItem()==null

  || jComboBox2.getSelectedItem()==null

      || jComboBox3.getSelectedItem()==null

      || jComboBox4.getSelectedItem()==null
```

```
        || jList3.isSelectionEmpty()==true

        || jComboBox6.getSelectedItem()==null

        || jComboBox7.getSelectedItem()==null

        || jComboBox8.getSelectedItem()==null

        || jComboBox9.getSelectedItem()==null

        || jList2.isSelectionEmpty()==true

        || jList4.isSelectionEmpty()==true

        || jList5.isSelectionEmpty()==true

        || jComboBox10.getSelectedItem()==null

        || jmood1.isSelectionEmpty()==true || check==3  || check2==3

        || jTextField1.getText().trim().equalsIgnoreCase("")){

    JOptionPane.showMessageDialog(this,"Gemiste ola ta pedia");

            pks=false;

  }    else{

         String selectedItem =(String)jComboBox10.getSelectedItem();

        if (selectedItem.equalsIgnoreCase("Single")){

         if(j1.getSelectedItem()==null || j2.getSelectedItem()==null){

             JOptionPane.showMessageDialog(this,"Gemiste ola ta pedia");

        pks=false; }

         else{

pks=true; }
```

```
        }else if (selectedItem.equalsIgnoreCase("Familly")){

            if(j3.getSelectedItem()==null || j5.getSelectedItem()==null ||

j4.getSelectedItem()==null || j6.getSelectedItem()==null ||
jComboBox17.getSelectedItem()==null || jComboBox18.getSelectedItem()==null ||
jComboBox19.getSelectedItem()==null){

  JOptionPane.showMessageDialog(this,"Gemiste ola ta pedia");

   pks=false;  } else{pks=true;} }

//Παίρνουμε την ώρα και την ημερομηνία που θα γίνει η αποθηκευσει.Το τελικό όνομα
που θα δώσει ο χρηστής θα είναι συνοδευόμενο από την ώρα και την ημερομηνία.

  if(pks==true){

       DateFormat dateFormat = new SimpleDateFormat("dd-MM-yyyy_HH.mm.ss");

   Date date = new Date();

   String currentDate=dateFormat.format(date);

//Δημιουργούμε σύνδεση με την Exist μέσου κατάλληλου driver και αποθηκεύουμε στην
μεταβλητή τύπου Collection col την collection που θα χρησιμοποιήσουμε(στην
περίπτωση μας την Collection με όνομα Templates.

try {

String driver = "org.exist.xmldb.DatabaseImpl";

 Class cl = Class.forName(driver);

 Database database = (Database) cl.newInstance();

 DatabaseManager.registerDatabase(database);

Collection col

=DatabaseManager.getCollection("xmldb:exist://localhost:8080/exist/xmlrpc/db/Templates");
```

```
XPathQueryService service = (XPathQueryService)col.getService("XPathQueryService", "1.0");service.setProperty("indent", "yes");

//Δημιουργούμε Query μέσω της γλώσσας XQuery και το αποτέλεσμα αυτής μας δίνει όλο το περιεχόμενο του Exist_template.xml μας που βρίσκεται στην Exist.

  ResourceSet result = service.query("for $x in doc('Exist_template.xml') return $x");

ResourceIterator i = result.getIterator();

        while (i.hasMoreResources()) {

         r =i.nextResource();

        }

Collection colfinal=DatabaseManager.getCollection("xmldb:exist://localhost:8080/exist/xmlrpc/db/final","admin","admin");

//Μέσω ενός XMLSerializer αποθηκεύουμε το περιεχόμενο του Exist_template.xml στο δίσκο μας.Ταυτοχρώνος δημιουργούμε σε καινούργιο φάκελο στην Exist ένα καινούργιο Document που είναι τύπου XMLResource και εκεί μέσα αποθηκεύουμε το παρεχόμενο που πήραμε από το Exist_template.xml.

XMLResource documentbef = (XMLResource)r;

//Μέσω ενός XMLSerializer αποθηκεύουμε το περιεχόμενο του Exist_template.xml στο δίσκο μας.

doc2=(Document) documentbef.getContentAsDOM();

doc2.normalize();

OutputFormat format = new OutputFormat(doc2);

format.setIndenting(true);

XMLSerializer serializer = new XMLSerializer(
```

```
new FileOutputStream(new File("C:\\Configuration\\XmlCopy.xml")), format);

serializer.serialize(doc2);

String onoma=jTextField1.getText();

String FinalOnoma=onoma+"_"+currentDate;

XMLResource

document=(XMLResource)colfinal.createResource(FinalOnoma,"XMLResource");

XPathQueryServiceservicefinal

= (XPathQueryService)colfinal.getService("XPathQueryService", "1.0");

        servicefinal.setProperty("indent", "yes");

File f2=new File("C:\\Configuration\\XmlCopy.xml");

if(!f2.canRead()) {

  System.out.println("cannot read file");

 return;

 }document.setContent(f2);

 System.out.println(document.getId());

colfinal.storeResource(document);

System.out.println("ok.");
```

//Σ'αυτό το σημείο παίρνουμε τους codes από τα τις προτιμησεις του χρηστή και τους αποθηκεύουμε σε συγκεκριμένες μεταβλητές πριν τα αποθηκεύσουμε στο τελικό Xml.Ολα τα δεδομένα των jComboBox και jList μας βρίσκονται σε μίκρα Xml που είναι αποθηκευμένα στο δίσκο μας στο σημείο C:\Configuration.

Στα atributes των επιλογών των κάθε Xml χωριστά βρίσκονται οι codes οι όποιοι χρειαζόμαστε..Πρίν γίνει η αντιστοίχηση των κωδικών με τις επιλογές που έκανε ο χρήστης αποθηκεύουμε σε μια μεταβλητή τύπου String την επιλογή που κάνει ο χρηστής.

```
minorFunc=(String)jComboBox2.getSelectedItem();
```

//Διαβάζουμε το Xml που θέλουμε από το δίσκο χρησιμοποιώντας έναν File Reader.Έπειτα αποθηκεύουμε το αποτέλεσμα από αυτά που διαβάσαμε σε μια μεταβλητή τύπου String.

```
File minFunc = new File("C:\\Configuration\\MinorRoom.xml");

FileReader rdminor = new FileReader(minFunc);

char[] bufminor = new char[(int)minFunc.length()];

 rdminor.read(bufminor);

  patternStr = new String(bufminor);
```

//Περνάμε τα δεδομένα σε μια μεταβλητή τύπου InputStream και δημιουργούμε ένα αντικείμενο τύπου DocumentBuilderFactory.Μέσω αυτού δημιουργούμε ένα αντικείμενο τύπου docBuilder και μεσω αυτού κάνουμε parse τα δεδομένα μας σε ένα αντικείμενο τύπου Document και με όνομα myXML.

```
InputStream isminor = new ByteArrayInputStream(patternStr.getBytes("utf-8"));

DocumentBuilderFactory docBuilderFactoryminor

= DocumentBuilderFactory.newInstance();

DocumentBuilder docBuilderminor =docBuilderFactoryminor.newDocumentBuilder();

Document myXMLminor=docBuilderminor.parse(isminor);

myXMLminor.normalize();
```

```
//Μεσω του myXML παίρνουμε το DocumentElement του XML μας και το αποθηκεύουμε στην μεταβλητή myXMLElement.Έπειτα μέσω αυτού παίρνουμε μια λίστα με τα Elements που έχουν tagName Item και τα αποθηκεύουμε σε μια μια μεταβλητή τύπου Nodelist

Element myXMLElementminor = myXMLminor.getDocumentElement();

  NodeList commonListminor = myXMLElementminor.getElementsByTagName("Item");

int listminor=0;

        while(listminor<commonListminor.getLength()){

      Node commonNode = commonListminor.item(listminor);

      Element commonElement = (Element)commonNode;

           Node myNode = commonNode.getNextSibling();

          myNode=commonElement.getFirstChild();

//Μέσω μιας επανάληψης While παίρνουμε μια μια τις τιμές από το Xml και τις συγκρίνουμε με την τιμή που έχουμε αποθηκεύσει στην μεταβλητη τύπου String.Όταν θα βρούμε την τιμή η οποία θα επαληθεύει την ισότητα παίρνουμε την τιμή του attribute που αντιστοιχεί σ'αυτήν (που αποτελεί και τον code που θέλουμε) και τον αποθηκεύουμε σε μια μεταβλητή τύπου String.Αξίζει να σημειωθεί ότι αυτός ο κώδικας επαναλαμβάνεται για όλα τα jComboBox και για όλα τα jList των οποίων χρειαζόμαστε τους codes τους.

                String a =myNode.getNodeValue();

           if(a.equalsIgnoreCase(minorFunc)){

               minorCode =commonElement.getAttribute("id");

                break;

          }else{              listminor++;    } }
```

//Στον δίσκο μας και στο path C:\Configuration υπάρχει ένα ακόμα Xml που ονομάζεται Rules.xml.Σ'αυτό το Xml είναι αποτυπωμένοι κάποιοι κανόνες που χρησιμοποιούνται για το γέμισμα του συγκεκριμένου χώρου.Κάθε κανόνας χωριστά εισάγει διαφορετικά στοιχεία μέσα στο χώρο.Στο attribute (με όνομα id) του κάθε κανόνα υπάρχει ένα 5ψηφιος αριθμος.Αυτός ο αριθμός δεν είναι τπτ άλλο από τα codes των επιλογών που επηρεάζουν το κάθε κανόνα.

//Ετσι αυτό που κάνουμε σ'αυτό το σημείο του κώδικα είναι να ελέγχουμε αν οι επιλογές που έχει κάνει ο χρηστής ταιριάζουν με κάποιο κανονα.Οι επιλογές του χρηστή που μας ενδιαφέρουν σε πρώτη φάση είναι το Room, Is this room going to be used only as"R1"or also as,State,NumberChildren,Character.Έτσι βάζουμε τα codes από αυτά τα στοιχεία σε ένα πινάκα String πέντε χαρακτήρων για να μπορέσουμε να τα συγκρίνουμε με τον attribute του κανόνα που και αυτό αποτελείτε από 5 χαρακτήρες.

```
int secXml=0;

        String aString = Integer.toString(secXml);

    rules[0]=majorCode;

    rules[1]=minorCode;

    rules[2]=stateCode;
```

//Σ'αυτό το σημείο ελένγχουμε τι έχει επιλέξει ο χρηστής όσο αναφορά την οικογενειακή του κατάσταση.Αν έχει επιλέξει Single τότε δίνουμε εμείς την τιμή 0 στον code που αφορά τον αριθμό των παιδιών του γιατί αλλιώς θα πάρει την τιμή null κάτι πού δεν θα μας βολεύει στην σύγκριση.

```
    if(selectedItem.equalsIgnoreCase("Single")){

        rules[3]=aString;

        kidsCodeRule=aString;

    }else if(selectedItem.equalsIgnoreCase("Familly"))

    {
```

```
   rules[3]=nokidsCode;

   kidsCodeRule=nokidsCode; }

rules[4]=characterCode;

Vector finalVector=new Vector();

//Οπως και στα προηγούμενα Xml διαβάζουμε το Xml από το δίσκο που έχει τα δεδομένα μας χρησιμοποιώντας έναν FileReader.Έπειτα αποθηκεύουμε το αποτέλεσμα από αυτά που διαβάσαμε σε μια μεταβλητή τύπου String.

File frules = new File("C:\\Configuration\\Rules.xml");

       FileReader rdrules = new FileReader(frules);

       char[] bufrules = new char[(int)frules.length()];

       rdrules.read(bufrules);

patternStr = new String(bufrules);

 //Περνάμε τα δεδομένα σε μια μεταβλητή τύπου InputStream και δημιουργούμε ένα αντικείμενο τύπου DocumentBuilderFactory.Μέσω αυτού δημιουργούμε ένα αντικείμενο τύπου docBuilder και μέσω αυτού κάνουμε parse τα δεδομένα μας σε ένα αντικείμενο τύπου Document και με όνομα myXML

InputStream isrules = new ByteArrayInputStream(patternStr.getBytes("utf-8"));

  DocumentBuilderFactory docBuilderFactoryrules

= DocumentBuilderFactory.newInstance();

DocumentBuilder docBuilderrules =docBuilderFactoryrules.newDocumentBuilder();

             Document myXMLrules =docBuilderrules.parse(isrules);

             myXMLrules.normalize();

         Element myXMLElementrules = myXMLrules.getDocumentElement();
```

```
NodeList commonList = myXMLElementrules.getElementsByTagName("rule");

   int irules=0;

        while (irules<commonList.getLength())

         {

    Node commonNode = commonList.item(irules);

        Element commonElement = (Element)commonNode;

  String b=(String)commonElement.getAttribute("id");
```

//Ο αλγόριθμος που ακολουθει αφορά τον έλεγχο που κάνουμε στο αν οι codes των επιλογών του χρηστή που αφορούν τους κανόνες είναι ίδιοι με κάποιο κανόνα που βρίσκεται στο xml με όνομα Rules.xml.Παίρνουμε ένα ένα τον code του κανόνα και βάζουμε ένα ένα τα στοιχεία του σε ένα πινάκα String με όνομα wrules.Έπειτα ελέγχουμε ένα ένα τα στοιχεία μεταξύ των δυο String εάν είναι ακριβώς ιδία μεταξύ τους.

```
 int attrlenght=0;

          int attrlenghtfin=1;

         String []wrules=new String[5];

          while(attrlenght<b.length()){

        wrules[attrlenght]=b.substring(attrlenght,attrlenghtfin);

             attrlenght++;

            attrlenghtfin++;

          }
```

//Από το σύστημα μας όταν ένας χρηστής επιλέγει ότι έχει οικογένεια τότε πρέπει να επιλέξει από ποσά παιδιά αποτελείτε η οικογένεια του.Κάποιοι όμως κανόνες αφορούν ανθρώπους που έχουν οικογένεια αλλά δεν έχουν κανένα παιδι.Έτσι αν βρούμε κάποιον τέτοιο κανόνα κάνουμε τον code που αφορά τον αριθμό των παιδιών ίσον με 0 για να διευκολύνουμε τον έλεγχο.

```
            if((wrules[2]).equalsIgnoreCase("2") && wrules[3].equalsIgnoreCase(aString)){
  rules[3]=aString;}
```

//Ξεκιναει ο έλεγχος..

```
int comp=0;
   While (comp<rules. Length) {
      if(rules[comp].equalsIgnoreCase(wrules[comp])){
                  comp++;
               khj=true;
               }else{
               khj=false;
               break;}}
```

//Αν οι επιλογές που έχει κάνει ο χρηστής επαληθεύουν κάποιο κανόνα μας τότε η μεταβλητή khj γίνεται true.Έπειτα μπαίνουμε στον κόμβο του κανόνα που επαλήθευσε τον έλεγχο μας και αποθηκεύουμε διάφορα στοιχεία του σε κάποιες μεταβλητές.Ελενχουμε ένα ένα τα παιδιά του κομβου.Αρχικά αποθηκεύουμε στην μεταβλητή nam το στοιχειό που θα επηρεάζεται από τον κανόνα και έπειτα στην μεταβλητή path το path στο οποίο βρίσκεται αυτό το στοιχειό στο Xml με όνομα XML_templates.xml.Αυτο το Xml είναι άδειο από προτιμήσεις ενός χρήστη αλλά έχει αποθηκευμένες όλες εκείνες τις πληροφορίες που χρειάζονται για την δομή ενός χώρου.Αποτελεί στην ουσία το template ενός δωματίου.Τέλος αποθηκεύουμε στον finalVector τις τιμές των attribute του στοιχείου που θα επηρεαστεί από τον κανόνα.

```
 if(khj==true){

 Node aaa=commonElement.getFirstChild().getNextSibling();

Element zzz=(Element)aaa;

 int num=0;

  while(zzz.hasAttributes()){

String attr=zzz.getAttribute("type");

   path=zzz.getAttribute("path");

   nam=zzz.getTagName();

  NodeList cv=commonElement.getElementsByTagName(nam);

   int countElem=cv.getLength();

        int yyy=0;

        while(yyy<countElem){

           Node nh=cv.item(yyy);

           Element cb=(Element)nh;

           attr=cb.getAttribute("type");

           finalVector.add(yyy, attr);

           yyy++; }
```

// Δημιουργούμε query στο 'XML_templates.xml' με βάση την μεταβλητή nam στην οποία έχουμε αποθηκεύσει το όνομα του στοιχείου που επηρεάζεται από τον κανονα.Το αποτέλεσμα της θα μας επιστρέψει όλο το κόμβο με το όνομα που έχουμε αποθηκεύσει στην μεταβλητή nam εκτός από τα στοιχεία στην opening tag και τα στοιχεία της closing tag.Στα στοιχεία αυτά υπάρχει και ένα attribute για κάθε Element με όνομα type.

```
//Στην τιμή του attribute κάθε φορά θα προσθέτουμε την τιμή που θα έχουμε την συγκεκριμένη στιγμή στο finalVector.

ResourceSet resultDoFurn = service.query("doc('XML_templates.xml')//"+nam+"/(* except "+nam+")");

  ResourceIterator iDoFurn = resultDoFurn.getIterator();

     Vector zits=new Vector();

  int k=0;

while (iDoFurn.hasMoreResources()) {

      rDoFurn =iDoFurn.nextResource();

zits.add(k,(String)rDoFurn.getContent());

  k++;}

  int z=0;

     while(z<zits.size()){

        if(z==0){

           inm=(String)zits.get(z);}

        else{

            inm=inm+(String)zits.get(z); }

   z++;}

/// Μέσω της XqueryUpdate κάνουμε insert το αποτέλεσμα της προηγούμενης Query μας στο Xml που θα αποτυπώσουμε τις επιθυμίες του χρηστή και ως path θα χρησιμοποιήσουμε την μεταβλητή path στην οποία έχουμε αποθηκεύσει το path που βρίσκεται το στοιχειό που επηρεάζεται από το κανόνα. Στην τιμή του attribute κάθε φορά θα προσθέτουμε την τιμή που θα έχουμε την συγκεκριμένη στιγμή στο finalVector.
```

```
ResourceSet copyFurn = servicefinal.query("update insert <"+nam+" type="'+finalVector.get(num)+"'>"+inm+"</"+nam+">into doc("'+FinalOnoma+"')/"+path+"['"+num+"']");

 num++;

  Node finalNode =zzz.getNextSibling();

//Για να μην μπεί ο η επανάληψη μας στην Closing Tag και μας επιστρέψει την τιμή null ελέγχουμε πότε το τελευταίο Element του DocumentElement (commonElement) είναι το ίδιο Element με το τελευταίο Element του κόμβου μας.

//Μολις συμβεί κάτι τέτοιο σταματάμε τον έλεγχο μας.

if(finalNode.isSameNode(commonElement.getLastChild())){

       break;}

finalNode= finalNode.getNextSibling();

 zzz=(Element) finalNode;

  } break;

  }else if(khj==false){

 irules++;  }

 //Αν βγει από την επανάληψη και η τιμή khj έχει την τιμή false(αυτό σημαίνει ότι κανένας κανόνας δεν ταίριαζε ακριβώς με τα codes των επιλογών του χρήστη )τότε κάνουμε ξανά τον ίδιο έλεγχο με την μόνη διαφορά αυτήν την φορά ότι θεωρούμε σαν ισότητα μεταξύ των δυο στοιχείων αν στον code που παίρνουμε από το Xml με τους κανόνες έχεις ως τιμή το 0.Έπειτα αφού βρούμε τον code που επαληθεύει τον κανόνα μας κάνουμε ακριβώς την ιδία διαδικασία με παραπάνω.

if(khj==false){

       irules=0;
```

```
 while (irules<commonList.getLength())

              {

Node commonNode = commonList.item(irules);

               Element commonElement = (Element)commonNode;

           String b=(String)commonElement.getAttribute("id");

               int attrlenght=0;

                int attrlenghtfin=1;

             String []wrules=new String[5];

               while(attrlenght<b.length()){

              wrules[attrlenght]=b.substring(attrlenght,attrlenghtfin);

                    attrlenght++;

                  attrlenghtfin++;}

               if((wrules[2]).equalsIgnoreCase("2") && wrules[3].equalsIgnoreCase(aString)){

                  rules[3]=aString;}

    int comp=0;

               while(comp<rules.length){

 if(rules[comp].equalsIgnoreCase(wrules[comp])||wrules[comp].equalsIgnoreCase(aString)){

                       comp++;

                    khj=true;

                    }else{

                   khj=false;
```

```
                    break;}}
 if(khj==true){
  Node aaa=commonElement.getFirstChild().getNextSibling();
 Element zzz=(Element)aaa;
 int num=0;
  while(zzz.hasAttributes()){
   String attr=zzz.getAttribute("type");
   path=zzz.getAttribute("path");
   nam=zzz.getTagName();
  NodeList cv=commonElement.getElementsByTagName(nam);
   int countElem=cv.getLength();
         int yyy=0;
         while(yyy<countElem){
            Node nh=cv.item(yyy);
            Element cb=(Element)nh;
            attr=cb.getAttribute("type");
            finalVector.add(yyy, attr);
            yyy++; }
ResourceSet resultDoFurn = service.query("doc('XML_templates.xml')//"+nam+"/(* except
"+nam+")");
 ResourceIterator iDoFurn = resultDoFurn.getIterator();
```

```
    Vector zits=new Vector();

 int k=0;

while (iDoFurn.hasMoreResources()) {

      rDoFurn =iDoFurn.nextResource();

zits.add(k,(String)rDoFurn.getContent());

 k++;}

 int z=0;

    while(z<zits.size()){

        if(z==0){

           inm=(String)zits.get(z);}

        else{

            inm=inm+(String)zits.get(z); }

  z++;}

ResourceSet copyFurn = servicefinal.query("update insert <"+nam+"
type="'+finalVector.get(num)+"'>"+inm+"</"+nam+">into
doc("'+FinalOnoma+"')/"+path+"['"+num+"']");

 num++;

 Node finalNode =zzz.getNextSibling();

if(finalNode.isSameNode(commonElement.getLastChild())){

   System.out.println("telos"); break;}

finalNode= finalNode.getNextSibling();
```

```
zzz=(Element) finalNode;

 } break;

 }else if(khj==false){irules++;  }
```

//Εκτός από τους παραπάνω κανόνες που επηρεάζουν τα διάφορα επιπλέον στοιχεία που θα προσθέσαμε στο χώρο έχουμε και τους κανόνες που επηρεάζουν τα χρώματα που θα έχει ο έχει ο χώρος.Το xml που είναι αποτυπωμένοι οι κανόνες για τα χρώματα βρίσκεται στο C:\Configuration με όνομα ColorRules.xml.Στο attribute (με όνομα id) του κάθε κανόνα υπάρχει ένα 2ψηφιος αριθμός.

//Αυτός ο αριθμός δεν είναι τίποτα άλλο από τα codes των επίλογων που επηρεάζουν το κάθε κανονα.Ετσι αυτό που κάνουμε σ'αυτό το σημείο του κώδικα είναι να ελέγχουμε αν οι επιλογές που έχει κάνει ο χρηστής ταιριάζουν με κάποιο κανονα.Οι επιλογές του χρηστή που μας ενδιαφέρουν είναι το Mood,και το ColorToAvoid. Έτσι βάζουμε τα codes από αυτά τα στοιχεία σε ένα πινάκα String 2 χαρακτήρων για να μπορέσουμε να τα συγκρίνουμε με το attribute του κανόνα που και αυτό αποτελείτε από 2 χαρακτήρες επίσης.Αξίζει να σημειωθεί ότι για να μπορέσει ο χρηστής να επιλέξει το ColorToAvoid θα πρέπει πρώτα να επιλέξει μία επιλογή στο Mood.Ανάλογα με την επιλογή που θα κάνει θα είναι και διαφορετική η τριπλέτα των χρωμάτων που θα εμφανίζεται στο ColorToAvoid.Όποιο Color επιλέξει ο χρηστής από την τριπλέτα χρωμάτων τα αλλά δυο θα είναι αυτά που θα αποτυπωθούν στο Xml στο στοιχειό με όνομα ColorPalette.

```
cPalette[0]=moodCode;

   cPalette[1]=cAvoidCode;

 Vector finalVectorcPallete=new Vector();

  File fcPalette = new File("C:\\Configuration\\ColorRules.xml");

      FileReader rdcPalette = new FileReader(fcPalette);

      char[] bufcPalette = new char[(int)fcPalette.length()];

      rdcPalette.read(bufcPalette);

      patternStr = new String(bufcPalette);
```

```
   InputStream iscPalette = new ByteArrayInputStream(patternStr.getBytes("utf-8"));
 DocumentBuilderFactory docBuilderFactorycPalette
= DocumentBuilderFactory.newInstance();
DocumentBuilder docBuildercPalette =docBuilderFactorycPalette.newDocumentBuilder();
   Document myXMLcPalette =docBuildercPalette.parse(iscPalette);
    myXMLcPalette.normalize();
 Element myXMLElementcPalette = myXMLcPalette.getDocumentElement();
 NodeList commonListcPalette = myXMLElementcPalette.getElementsByTagName("rule");
      int icPalette=0;
   while (icPalette<commonListcPalette.getLength(){
  Node commonNode = commonListcPalette.item(icPalette);
              Element commonElement = (Element)commonNode;
           String b=(String)commonElement.getAttribute("id");
              int attrlenght=0;
               int attrlenghtfin=1;
         String []wcPalette=new String[2];
              while(attrlenght<b.length()){
 wcPalette[attrlenght]=b.substring(attrlenght,attrlenghtfin);
                 attrlenght++;
                 attrlenghtfin++; }
                int comp=0;
```

```
            while(comp<cPalette.length){

     if(cPalette[comp].equalsIgnoreCase(wcPalette[comp])){

                    comp++;

                    khj=true;}else{

                    khj=false;

                    break;}}

   f(khj==true){

 NodeList  nodeList1=commonElement.getElementsByTagName("ColorPalette");

         int k=0;

    while (k<nodeList1.getLength()){

        Node node1 = nodeList1.item(k);

              Element element1 = (Element)node1;

         Node myNode = node1.getNextSibling();

          myNode=element1.getFirstChild();

             String a =myNode.getNodeValue();

         finalVectorcPallete.add(k,a);

         k++;

}

          break;

          }else if(khj==false){ icPalette++;    }}
```

```
//Μολις τελειώσει η επανάληψη στο Vector finalVectorcPallete έχουμε αποθηκεύσει τα 2 χρώματα που θα χρειαστεί να κάνουμε insert στο Xml.Αυτό γίνεται στο παρακάτω κώδικα.

int  uy=0;

  while(uy<finalVectorcPallete.size()){

ResourceSet colourPallete=servicefinal.query("update replace doc('"+FinalOnoma+"') //ColorPalette with <ColorPalette ColorPaletteIndexName='default ColorPalette Index Names' ColorPaletteIndexURL='"+pathCAvoid+"'>"+finalVectorcPallete.get(uy)+"</ColorPalette>") ; uy++;}

//Στο παρακάτω String αποθηκεύουμε τους codes που επηρεάζουν τους κανόνες που είναι υπεύθυνοι για το γέμισμα του συγκεκριμένου χώρου με επιπλέον πραγματα.Αυτή η τιμή θα μας βοηθήσει στην εκτέλεση του Update.

 String resultRule;

resultRule=majorCode+minorCode+stateCode+kidsCodeRule+characterCode;

//Το παρακάτω module κώδικα αφορά το χρώμα που επιλέγει στο κουμπί με όνομα FavouriteColour.Παίρνει το χρώμα και το μετατρέπουμε σε δεκαεξαδικο.

Color c1=jButton5.getBackground();

 String red = Integer.toHexString(c1.getRed());

String green = Integer.toHexString(c1.getGreen());

String blue = Integer.toHexString(c1.getBlue());

 if (red.length() == 1)

red = "0" + red;

 if (green.length() == 1)  green = "0" + green;
```

```
   if (blue.length() == 1)

        blue = "0" + blue;

 String hexColorb1 = "#" + red + green + blue;

//Σ'αυτό το σημείο πηγαίνουμε και αποθηκεύουμε στο Xml τις προτιμήσεις και τις επιθυμίες του χρηστή.

 ResourceSet roomMajor=servicefinal.query("update replace doc('"+FinalOnoma+"') //MajorFunctionality with

<MajorFunctionality MajorFunctionalityIndexName='default MajorFunctionalityIndeNames' MajorFunctionalityIndexURL='+pathMajor+'> "+majorCode+" MajorFunctionality>")

//Αξιζει να σημειωθεί ότι αυτός ο κώδικας επαναλαμβάνεται για όλες τις επιθυμίες του χρηστή.Τις μεταβλητές που δίνουν τιμη στο καθε IndexURL τις παίρνουμε από το Combobox.xml και αυτές οι μεταβλητες δηλώνουν το path που βρίσκεται το κάθε xml με τα δεδομένα του κάθε jCombobox και Jlist στο δίσκο.

//Αυτός ο κώδικας επαναλαμβάνεται για όλα τα Combobox και Jlist που ο χρηστής αποτυπώνει τις επιθυμίες και χρειάζονται για το γέμισμα του Xml.

// Ο κωδικας που μας δινει το path

NodeList listMajorPath = myXMLElement.getElementsByTagName("MajorRoom");

      Node nodeMajor = listMajorPath.item(0);

      Element elementMajor = (Element)nodeMajor;

       Node myNodeMajor = nodeMajor.getNextSibling();

         myNodeMajor=elementMajor.getFirstChild();

       String pathMajor =myNodeMajor.getNodeValue();
```

3.4 Ανάλυση της κλάσης MiniBrowser

3.4.1 Επεξήγηση της μεθόδου doStyle()

```
private void doStyle() {

//Διαβάζουμε το Xml από το δίσκο που έχει τα δεδομένα μας χρησιμοποιώντας έναν FileReader.Έπειτα αποθηκεύουμε το αποτέλεσμα από αυτά που διαβάσαμε σε μια μεταβλητή τύπου String.

try{

File f = new File("C:\\Configuration\\BrowserStyles.xml");

        FileReader rd = new FileReader(f);

        char[] buf = new char[(int)f.length()];

        rd.read(buf);

       patternStr = new String(buf);

  patternStr=patternStr.substring(3);

//Περνάμε τα δεδομένα σε μια μεταβλητή τύπου Input Stream και δημιουργούμε ένα αντικείμενο τύπου DocumentBuilderFactory.Μέσω αυτού δημιουργούμε ένα αντικείμενο τύπου docBuilder και μέσω αυτού κάνουμε parse τα δεδομένα μας σε ένα αντικείμενο τύπου Document και με όνομα myXML.

InputStream is = new ByteArrayInputStream(patternStr.getBytes("utf-8"));

  DocumentBuilderFactory

  docBuilderFactory= DocumentBuilderFactory.newInstance();

 DocumentBuilder docBuilder =docBuilderFactory.newDocumentBuilder();

   Document myXML =(Document) docBuilder.parse(is);

    myXML.normalize();
```

```
   Element myXMLElement = myXML.getDocumentElement();

//Συγκρίνουμε την μεταβλητή rs που την είχαμε στείλει σαν όρισμα μέσα από την κλάση
MyWizard με την μεταβλητή τύπου String με όνομα Modern(είναι ένα από τα ονόματα
των Style που έχουμε).Αν ισχύει η ισότητα φτάνουμε μέχρι την τιμή του πρώτου παιδιού
του κάθε Element και την αποθηκεύουμε σε μια μεταβλητή τύπου String με όνομα
finalUrl.

  if(rs.equalsIgnoreCase("Modern")){

   NodeList commonList = myXMLElement.getElementsByTagName("Modern");

   Node commonNode = commonList.item(0);

  Element commonElement = (Element)commonNode;

   Node myNode = commonNode.getNextSibling();

   myNode=commonElement.getFirstChild();

  finalUrl=myNode.getNodeValue();

//Στο locationTextField του Browser δίνουμε κάθε φορά την τιμή της finalUrl.

    locationTextField.setText(finalUrl);

//Με την loadPage() φορτώνουμε κάθε φορά την ανάλογη σελίδα.

    loadPage();  }  }

    catch(Exception e){e.printStackTrace();} }

//Αξίζει να σημειωθεί ότι αυτός ο κώδικας επαναλαμβάνεται για κάθε style
χωριστα.Δηλαδη στην θέση του Modern έχουμε επίσης και Classic, Retro, African,
Cottage, Futuristic, Japanese, Minimal,Morocco.Μ'αυτήν την μέθοδο πετυχαίνουμε να
ελέγχουμε ανάλογα με το style που έχουμε επιλέξει από την κλάση MyWizard πιο θα
είναι και το site με τις φωτογραφίες που θα σηκώσει η εφαρμογή μας.(MiniBrowser).Οι
τοποθεσίες των site βρίσκονται στο ιεραρχικό xml που είναι αποθηκευμένο στο δίσκο
μας με όνομα BrowserStyles.xml
```

3.4.2 Επεξήγηση της μεθόδου LoadPage()

```
private void loadPage()

//Σ'αυτήν την μέθοδο παίρνουμε την διεύθυνση από την address bar και την περνάμε σε μια μεταβλητή τύπου String.Δημιουργούμε ένα νέο αντικείμενο τύπου Url και περνάμε σαν όρισμα την μεταβλητή τύπου String.Έπειτα καλούμε την μέθοδο showPage περνώντας την σαν όρισμα το αντικείμενο τύπου Url.

{ try{

 String s = locationTextField.getText();

  URL u = new URL(s);

            showPage(u);

            }  catch (Exception e){} }
```

3.4.3 Επεξήγηση της μεθόδου showPage()

```
private void showPage(URL pageUrl){

//Αυτή η μέθοδος είναι υπεύθυνη για την εμφάνιση της ιστοσελίδας μας στον JEditorPane.Στέλνουμε με μορφή Url το link που βρισκεται στην addressbar του Browser και μέσω της εντολής displayEditorPane.setPage(pageUrl) η σελίδα που επιθυμούμε εμφανίζεται στον Browser.

setCursor(Cursor.getPredefinedCursor(Cursor.WAIT_CURSOR));

try {URL currentUrl = displayEditorPane.getPage();

displayEditorPane.setPage(pageUrl);

//Αν πατήσουμε κάποιο άλλο link ενημερώνουμε την address Bar με το καινούργιο Url.

URL newUrl = displayEditorPane.getPage();
```

```
locationTextField.setText(newUrl.toString());}

//Αν ο browser δεν θα μπορέσει να εμφανίσει για κάποιο λόγο την σελίδα που θέλουμε μας ενημερώνει με το παρακάτω μήνυμα.

catch (Exception e){

showError("Unable to load page");}

finally{setCursor(Cursor.getDefaultCursor());}}
```

3.5 Ανάλυση της Κλάσης Update.java

Η κλάση Update.java είναι ιδία με την κλάση MyWizard εκτός από την μέθοδο updateForm() και την μέθοδο ActionPerformed() του κουμπιού jButton4(Update) στο σημείο που θα γίνουν Update τα στοιχεία που παίρνουμε από τους κανόνες και συγκεκριμένα στο σημείο που μετράμε τα στοιχεία από τους κανόνες που έχουμε ήδη στο Xml και στα στοιχεία από τους κανόνες που θα αποτυπωθούν στο Xml μετά την διαδικασία του Update .Η μέθοδος updateForm() καλείτε μέσα από τον Constructor και είναι υπεύθυνη για να 'σηκώσει' τις επιλογές του χρηστή που έχουν αποτυπωθεί στο Xml (οι περισσότερες σε μορφή code)και να τις αντιστοιχήσει με τις αντιστοιχείς τιμές τους στα ξεχωριστά xml.Στα ξεχωριστά Xml στην θέση της τιμής του attribute έχουμε τον code και στην θέση του τιμής του πρώτου παιδιού για κάθε Elemet έχουμε την τιμή που αντιστοιχεί στον code.Αξίζει να σημειωθεί ότι στον Constructor της κλάσης Update παίρναμε σαν όρισμα το όνομα του Xml(σε μια μεταβλητή τύπου String) που έχει αποτυπωμένο τις επιθυμίες του χρηστή και θα είναι αυτό στο οποίο θα γίνει το Update.

3.5.1 Επεξήγηση της μεθόδου UpdateForm()

```
//Κάνουμε σύνδεση με την Exist και παίρνουμε το Collection στο οποίο έχουμε αποθηκεύσει το Xml στο οποίο είναι αποτυπωμένες οι επιθυμίες του χρηστή.

try{

  String driver = "org.exist.xmldb.DatabaseImpl";

   Class cl = Class.forName(driver);
```

```
    Database database = (Database) cl.newInstance();

     DatabaseManager.registerDatabase(database);

Collection col =
DatabaseManager.getCollection("xmldb:exist://localhost:8080/exist/xmlrpc/db/final");

    XPathQueryService service = (XPathQueryService) col.getService("XPathQueryService",
"1.0");

   service.setProperty("indent", "yes");

//Δημιουργούμε Queries που το αποτέλεσμα τους θα μας δώσει την επιθυμία του χρηστή
αποτυπωμένη σε ένα κωδικό.Αυτό τον κωδικό τον αποθηκεύουμε σε μια μεταβλητή
τύπου String.

  ResourceSet style = service.query("data(doc('"+xml+"')//style)");

             m = style.getIterator();

        while(m.hasMoreResources()) {

            Resource r = m.nextResource();

            roomStyle=((String)r.getContent());  }

//Πηγαίνουμε στο xml που αντιστοιχεί στο JComboBox ή Jlist για το οποίο ψάχνουμε
την τιμή.Ελέχγουμε τον code που έχουμε κρατήσει στην μεταβλητή τύπου String από
πάνω με ποιά τιμή του attribute είναι ίδιος.

      File frstyle = new File("C:\\Configuration\\RoomStyle.xml");

        FileReader rdstyle = new FileReader(frstyle);

        char[] bufstyle = new char[(int)frstyle.length()];

        rdstyle.read(bufstyle);

        patternStr = new String(bufstyle);

 InputStream isstyle = new ByteArrayInputStream(patternStr.getBytes("utf-8"));

   DocumentBuilderFactory docBuilderFactorystyle =
DocumentBuilderFactory.newInstance();

DocumentBuilder docBuilderstyle =docBuilderFactorystyle.newDocumentBuilder();


Document myXMLstyle=docBuilderstyle.parse(isstyle);

        myXMLstyle.normalize();
```

```
Element myXMLElementstyle = myXMLstyle.getDocumentElement();
    commonListstyle = myXMLElementstyle.getElementsByTagName("Item");
 int istyle=0;
         while(istyle<commonListstyle.getLength()){
         Node styleNode = commonListstyle.item(istyle);
       Element styleElement = (Element)styleNode;
          styleCode =styleElement.getAttribute("id");
//Μόλις βρούμε τον κόμβο που έχει ίδιο code παίρνουμε την τιμή του πρώτου παιδιού του και με την μέθοδο setSelectedItem ή με την μεθοδο setSelectedValue δίνουμε τιμές σε κάθε JComboBox  ή για κάθε JList αντίστοιχα.Αυτές οι τιμές είναι οι τιμές που έχει επιλέξει ο χρηστής κατά την διαδικασία της δημιουργίας νέου δωματίου και έτσι κατά της διαδικασίας του Update είναι οι και οι τιμές που θα εμφανίζονται στα JComboBox και JList.Αξίζει να σημειωθεί ότι όλη αυτήν η διαδικασία είναι ίδια για όλα τα JCoboBox και JList για τα οποία έχουμε αποθηκεύσει τα code τους στο Xml που είναι αποτυπωμένες οι επιθυμίες του χρηστή.
if(styleCode.equalsIgnoreCase(roomStyle)){
              Node myStyleNode = styleNode.getNextSibling();
         myStyleNode=styleElement.getFirstChild();
               styleUpdate =myStyleNode.getNodeValue();
              break;
              }else{
         istyle++;}}
     jComboBox3.setSelectedItem(styleUpdate);
```

3.5.2 Επεξήγηση της μεθόδου jButton4ActionPerformed ()(κουμπί Update)

```
//Δημιουργούμε Query που το αποτέλεσμα της μας γυρνά πόσα στοιχεία με  όνομα που έχει η μεταβλητή nam υπάρχουν αποτυπωμένα στο Xml.
 ResourceSet resultDoCount = servicefinal.query("count(doc('"+xml+"')//"+nam+")");
  ResourceIterator iDoCount = resultDoCount.getIterator();
   while (iDoCount.hasMoreResources()) {
```

```
  Resource rDoCount =iDoCount.nextResource();

 doCount=((String)rDoCount.getContent());

            }

     int count =Integer.parseInt(doCount);

//Αν ο αριθμός των στοιχείων που θα πάρουμε είναι 0  τότε κάνουμε insert του κάθε
στοιχείου όσε φορές είναι και ο αριθμός των Elements(countElem) που περιέχει ο
κόμβος με όνομα nam.

     if(count==0){

       int k=0;

     while(k<countElem){

  ResourceSet copyFurn = servicefinal.query("update insert <"+nam+"
type="'+finalVector.get(k)+'">"+inm+"</"+nam+"> into doc('"+xml+"')/"+path+"['"+k+"']");

     k++;}

//Εαν ο αριθμός των στοιχείων που θα πάρουμε είναι >0  τότε αφαιρούμε από τον αριθμό
των στοιχείων με  όνομα που έχει η μεταβλητή nam και υπάρχουν ήδη αποτυπωμένα στο
Xml τον αριθμό Elements(countElem) που περιέχει ο κόμβος που επαλήθευσε τον
κανόνα με όνομα nam.Το αποτέλεσμα το αποθηκεύουμε στην μεταβλητή compare.

}else if (count>0) {

 int compare=count-(countElem);

//Αν ο αριθμός που είναι αποτυπωμένος στην μεταβλητή compare είναι μεγαλύτερος από
0 τότε αυτό σημαίνει ότι ο αριθμός των στοιχείων με  όνομα που έχει η μεταβλητή nam
και υπάρχουν ήδη αποτυπωμένα στο Xml είναι μεγαλύτερος από τον αριθμό των
Elements που θα αποτυπωθούν στο Xml με το Update.

  if(compare>0){

       int k=0;

      int z9=1;

//Αρα θα χρειαστούμε να κάνουμε Update στα ήδη υπάρχον Elements οσάς φορές είναι ο
αριθμός countElem (αριθμός Elements που περιέχει ο κόμβος που επαλήθευσε τον
κανόνα με όνομα nam) και να κάνουμε Delete στοιχείων όσες φορές είναι ο αριθμός που
είναι αποτυπωμένος στην μεταβλητή compare (αριθμός που προήλθε από την παραπάνω
αφαίρεση)

      while(k<countElem){
```

```
ResourceSet Upd=servicefinal.query("update replace doc('"+xml+"')//"+nam+"["+z9+"] with
<"+nam+" type='"+finalVector.get(k)+"'>"+inm+"</"+nam+">");

      k++ ;z9++;}

     int m=0;

   while(m<compare){

ResourceSet del=servicefinal.query("update delete doc('"+xml+"')//"+nam+"[last()]");

  m++;

//Αν ο αριθμός που είναι αποτυπωμένος στην μεταβλητή compare είναι <0 τότε σημαίνει
ότι ο αριθμός των στοιχείων με  όνομα που έχει η μεταβλητή nam και υπάρχουν ήδη
αποτυπωμένα στο Xml είναι  μικρότερος από τον αριθμό των Elements που θα
αποτυπωθούν στο Xml μετά το Update.

 } }else if(compare<0){

//Λογω ότι αριθμός μετά την αφαίρεση θα βγει αρνητικός θα πάρουμε το απόλυτο της
τιμής.

       compare=java.lang.Math.abs(compare);

        int k=0;

        int z9=1;

//Αρα θα χρειαστούμε να κάνουμε Update στα ήδη υπάρχον Elements οσες φορές είναι ο
αριθμός count (αριθμός Elements που υπάρχει ήδη αποτυπωμένος στο Xml με όνομα
nam) και να κάνουμε Insert στοιχείων όσες φορές είναι ο αριθμός που είναι
αποτυπωμένος στην μεταβλητή compare (αριθμός που προηλθε από την παραπάνω
αφαίρεση)

while(k<count){

ResourceSet Upd=servicefinal.query("update replace doc('"+xml+"')//"+nam+"["+z9+"] with
<"+nam+" type='"+finalVector.get(k)+"'>"+inm+"</"+nam+">");

      k++;

       z9++;

   }

    int m=0;

    int vec=k-1;

   while(m<compare){
```

```
ResourceSet del=servicefinal.query("update insert
<"+nam+"type="'+finalVector.get(vec)+'">"+inm+"</"+nam+"> int
doc('"+xml+"')/"+path+"['"+k+"']");

  m++;  k++;

vec++;

 }

//Αν ο αριθμός που είναι αποτυπωμένος στην μεταβλητή compare είναι ίσος με 0 τότε
σημαίνει ότι ο αριθμός των στοιχείων με  όνομα που έχει η μεταβλητή name και
υπάρχουν ήδη αποτυπωμένα στο Xml είναι ίσος με τον αριθμό των Elements που θα
αποτυπωθούν στο Xml μετά το Update.

}else if (compare==0){

    int k=0;

    int z9=1;

//Άρα θα χρειαστούμε μόνο Update όσες φορές είναι και αριθμός Elements(countElem)
που περιέχει ο κόμβος που επαλήθευσε τον κανόνα με όνομα nam

       while(k<countElem){

ResourceSet Upd=servicefinal.query("update replace doc('"+xml+"')//"+nam+"["+z9+"] with
<"+nam+" type="'+finalVector.get(k)+'">"+inm+"</"+nam+">");

      k++;

      z9++;}} }
```

3.6 Ανάλυση της κλάσης Proxy.java

3.6.1 Επεξήγηση της μεθόδου doExit()

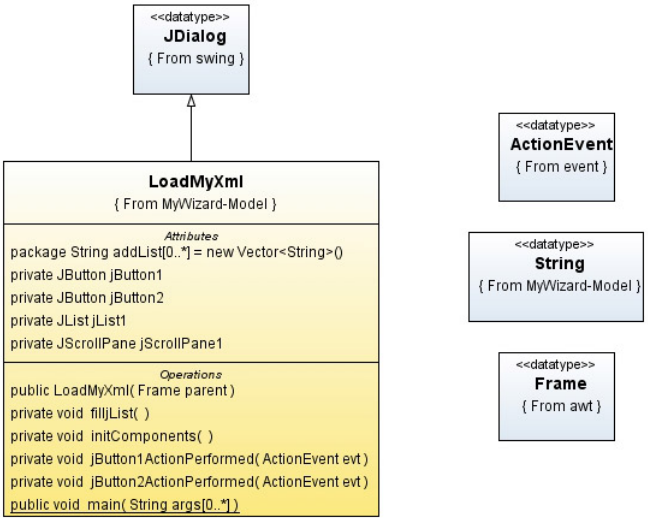
```
//Χρησιμοποιούμε την κλάση Proxy.java και πιο συγκεκριμένα την μέθοδο doExit() για
να δώσουμε αν χρειαστεί τιμές Host και Port στην εφαρμογή MiniBrowser.Το
αντικείμενο Proxy καλείτε μέσα από την εφαρμογή MiniBrowser εάν επιλέξουμε από το
JMenu την επιλογή ProxySettings.Αξίζει να σημειωθεί ότι οι τιμές που δίνουμε στο Host
και Port αποθηκευεται σε ένα Xml που ονομάζεται Browser.xml και βρίσκεται στο  path
C:\\Configuration.

private void doExit() {

      try{

            File f = new File("C:\\Configuration\\Browser.xml");
```

```
        FileReader rd = new FileReader(f);
        char[] buf = new char[(int)f.length()];
        rd.read(buf);
       String patternStr = new String(buf);
    InputStream is = new ByteArrayInputStream(patternStr.getBytes("utf-8"));
 DocumentBuilderFactory docBuilderFactory = DocumentBuilderFactory.newInstance();
              DocumentBuilder docBuilder =docBuilderFactory.newDocumentBuilder();
              Document myXML =docBuilder.parse(is);
              myXML.normalize();
               Element myXMLElement = myXML.getDocumentElement();
      NodeList commonList = myXMLElement.getElementsByTagName("Host");
              NodeList commonList1=myXMLElement.getElementsByTagName("Port");
            Node commonNode = commonList.item(0);
          Node commonNode1=commonList1.item(0);
          Element commonElement = (Element)commonNode;
           Element commonElement1=(Element)commonNode1;
//Ελέγχουμε στο Browser.xml αν έχει ήδη τιμες.Αν ναι κάνουμε replace τις υπάρχουσες
τιμές και αν όχι δημιουργούμε καινούργιους TextNodes και αποθηκεύουμε εκεί τις
τιμές.
                Boolean z= commonElement.hasChildNodes();
             Boolean m= commonElement1.hasChildNodes();
       if ((z==false) && (m==false)){
                Text S1 = myXML.createTextNode(jTextField1.getText());
                    commonElement.appendChild(S1);
             Text S2 = myXML.createTextNode(jTextField2.getText());
commonElement1.appendChild(S2);
                OutputFormat format = new OutputFormat(myXML);
                                format.setIndenting(true);
```

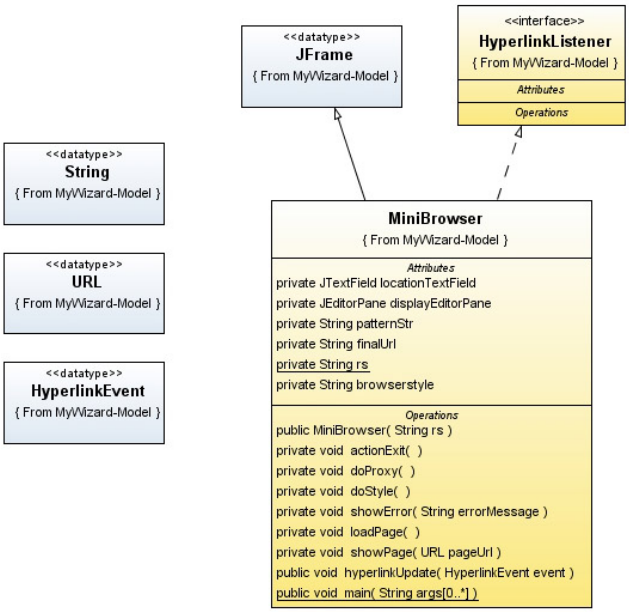
```
XMLSerializer serializer = new XMLSerializer(
            new FileOutputStream(new File("C:\\Configuration\\Browser.xml")), format);
                serializer.serialize(myXML);
          }
 else   if ((z==true) && (m==true)){
        Node myNode = commonNode.getNextSibling();
         Node myNode1 = commonNode1.getNextSibling();
         myNode=commonElement.getFirstChild();
          myNode1=commonElement1.getFirstChild();
              Text S1 = myXML.createTextNode(jTextField1.getText());
                commonElement.replaceChild(S1,myNode);
        Text S2 = myXML.createTextNode(jTextField2.getText());
                commonElement1.replaceChild(S2,myNode1);
//Αφού τελεύσουμε με την αποθήκευση κάνουμε serialize το xml Browser.xml μέσω ενός
XMLSerializer.
     OutputFormat format = new OutputFormat(myXML);
                              format.setIndenting(true);
                XMLSerializer serializer = new XMLSerializer(
            new FileOutputStream(new File("C:\\Configuration\\Browser.xml")), format);
                serializer.serialize(myXML);    } }
      catch(Exception e1){
         e1.printStackTrace();}
//Σαν τελευταίο βήμα δίνουμε στο σύστημα μας τις τιμές Host και  Port που έχουμε
γράψει στο  jTextField1.getText() και jTextField2.getText() μέσω της μεθόδου
System.getProperties().
System.getProperties().put("proxySet", "true");
            System.getProperties().put("proxyHost", jTextField1.getText());
            System.getProperties().put("proxyPort",jTextField2.getText());  }
```

3.7 Class Diagram

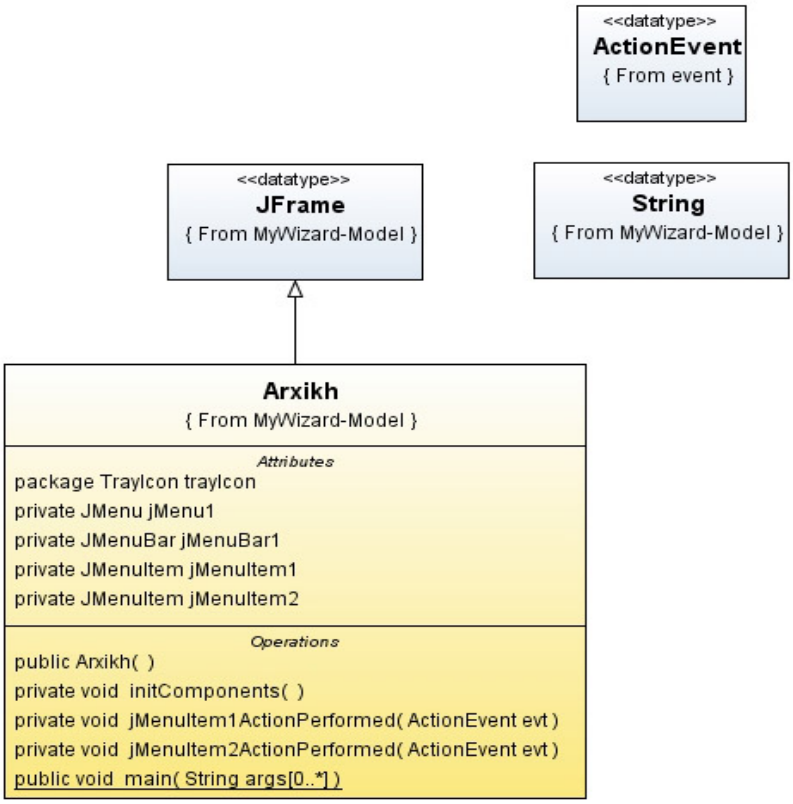
3.7.1 Class Diagram της κλάσης LoadMyXml.java



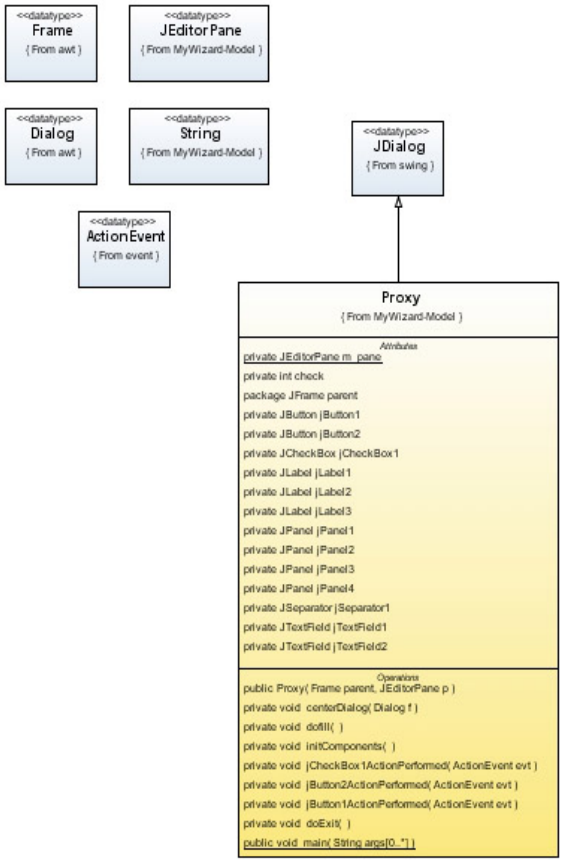
3.7.2 Class Diagram της κλάσης MiniBrowser.java



3.7.3 Class Diagram της κλάσης Arxikh.java



3.7.4 Class Diagram της κλάσης Proxy.java



4 ΒΙΒΛΙΟΓΡΑΦΙΑ

On Line:

http://www.w3schools.com/

http://www.w3.org/

http://exist-db.org/

http://www.it.uom.gr/project/xml/Home%20Page.htm

http://en.wikipedia.org/wiki/Main_Page

http://forums.devx.com/showthread.php?t=149187

Βιβλία:

OReilly.XQuery.Mar.2007 by Priscilla Walmsley